

1 INTERNET-DRAFT

R. Bergman
Dataproducts Corp.
T. Hastings
Xerox Corporation
S. Isaacson
Novell, Inc.
H. Lewis
IBM Corp.

~~February 3~~October 2, 1998

10 Job Monitoring MIB - V1.2
11 <draft-ietf-printmib-job-monitor-087.txt>
12

13 Status of this Memo

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

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

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

28 This Internet-Draft expires on ~~August 3~~April 2, 1998.

29
30

Abstract

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

45			
46		TABLE OF CONTENTS	
47	1.	INTRODUCTION	6
48	1.1	Types of Information in the MIB	6
49	1.2	Types of Job Monitoring Applications	8
50	2.	TERMINOLOGY AND JOB MODEL	9
51	2.1	System Configurations for the Job Monitoring MIB	12
52	2.1.1	Configuration 1 - client-printer	12
53	2.1.2	Configuration 2 - client-server-printer - agent in the	
54		server	13
55	2.1.3	Configuration 3 - client-server-printer - client monitors	
56		printer agent and server	14
57	3.	MANAGED OBJECT USAGE	16
58	3.1	Conformance Considerations	16
59	3.1.1	Conformance Terminology	16
60	3.1.2	Agent Conformance Requirements	16
61	3.1.2.1	MIB II System Group objects	17
62	3.1.2.2	MIB II Interface Group objects	17
63	3.1.2.3	Printer MIB objects	17
64	3.1.3	Job Monitoring Application Conformance Requirements	17
65	3.2	The Job Tables and the Oldest Active and Newest Active Indexes	18
66	3.3	The Attribute Mechanism	20
67	3.3.1	Conformance of Attribute Implementation	20
68	3.3.2	Useful, 'Unknown', and 'Other' Values for Objects and	
69		Attributes	21
70	3.3.3	Index Value Attributes	21
71	3.3.4	Data Sub-types and Attribute Naming Conventions	22
72	3.3.5	Single-Value (Row) Versus Multi-Value (MULTI-ROW)	
73		Attributes	22
74	3.3.6	Requested Objects and Attributes	23
75	3.3.7	Consumption Attributes	23
76	3.3.8	Attribute Specifications	23
77	3.4	Monitoring Job Progress	43
78	3.5	Job Identification	48
79	3.6	Internationalization Considerations	49
80	3.6.1	Text generated by the server or device	49
81	3.6.2	Text supplied by the job submitter	50
82	3.6.3	'DateAndTime' for representing the date and time	51

83	3.7 IANA and PWG Registration Considerations	51
84	3.7.1 PWG Registration of enums	52
85	3.7.1.1 Type 1 enumerations	52
86	3.7.1.2 Type 2 enumerations	52
87	3.7.1.3 Type 3 enumeration	53
88	3.7.2 PWG Registration of type 2 bit values	53
89	3.7.3 PWG Registration of Job Submission Id Formats	53
90	3.7.4 PWG Registration of MIME types/sub-types for document-	
91	formats 53	
92	3.8 Security Considerations	54
93	3.8.1 Read-Write objects	54
94	3.8.2 Read-Only Objects In Other User's Jobs	54
95	3.9 Notifications	54
96	4. MIB SPECIFICATION	54
97	Textual conventions for this MIB module	56
98	JmUTF8StringTC	56
99	JmJobStringTC	56
100	JmNaturalLanguageTagTC	56
101	JmTimeStampTC	57
102	JmJobSourcePlatformTypeTC	57
103	JmFinishingTC	59
104	JmPrintQualityTC	60
105	JmPrinterResolutionTC	60
106	JmTonerEconomyTC	61
107	JmBooleanTC	61
108	JmMediumTypeTC	61
109	JmJobCollationTypeTC	64
110	JmJobSubmissionIDTypeTC	64
111	JmJobStateTC	69
112	JmAttributeTypeTC	72
113	JmJobServiceTypesTC	76
114	JmJobStateReasons1TC	78
115	JmJobStateReasons2TC	82
116	JmJobStateReasons3TC	87
117	JmJobStateReasons4TC	87
118	The General Group (MANDATORY)	89
119	jmGeneralJobSetIndex (Int32(1..32767))	90
120	jmGeneralNumberOfActiveJobs (Int32(0..))	90
121	jmGeneralOldestActiveJobIndex (Int32(0..))	91
122	jmGeneralNewestActiveJobIndex (Int32(0..))	91
123	jmGeneralJobPersistence (Int32(15..))	92
124	jmGeneralAttributePersistence (Int32(15..))	92
125	jmGeneralJobSetName (UTF8String63)	93
126	The Job ID Group (MANDATORY)	93
127	jmJobSubmissionID (OCTET STRING(SIZE(48)))	95

128	jmJobIDJobSetIndex (Int32(0..32767))	96
129	jmJobIDJobIndex (Int32(0..))	96
130	The Job Group (MANDATORY)	96
131	jmJobIndex (Int32(1..))	99
132	jmJobState (JmJobStateTC)	99
133	jmJobStateReasons1 (JmJobStateReasons1TC)	100
134	jmNumberOfInterveningJobs (Int32(-2..))	100
135	jmJobKOctetsPerCopyRequested (Int32(-2..))	101
136	jmJobKOctetsProcessed (Int32(-2..))	101
137	jmJobImpressionsPerCopyRequested (Int32(-2..))	102
138	jmJobImpressionsCompleted (Int32(-2..))	102
139	jmJobOwner (JobString63)	103
140	The Attribute Group (MANDATORY)	103
141	jmAttributeTypeIndex (JmAttributeTypeTC)	106
142	jmAttributeInstanceIndex (Int32(1..32767))	106
143	jmAttributeValueAsInteger (Int32(-2..))	107
144	jmAttributeValueAsOctets (Octets63)	108
145	5. APPENDIX A - IMPLEMENTING THE JOB LIFE CYCLE	111
146	6. APPENDIX B - SUPPORT OF JOB SUBMISSION PROTOCOLS	112
147	7. REFERENCES	112
148	8. AUTHOR'S ADDRESSES	114
149	9. CHANGE HISTORY	116
150	9.1 Changes to produce version 1.1, dated October 1, 1998	116
151	<u>9.2 Changes to produce version 1.2, dated October 2, 1998</u>	117
152	10. INDEX	118
153		

154 Job Monitoring MIB

155 1 Introduction

156 This specification defines an official Printer Working Group (PWG)
157 [PWG] standard SNMP MIB for the monitoring of jobs on network printers.
158 This specification is being published as an IETF Information Document
159 for the convenience of the Internet community. In consultation with
160 the IETF Application Area Directors, it was concluded that this MIB
161 specification properly belongs as an Information document, because this
162 MIB monitors a service node on the network, rather than a network node
163 proper.

164 The Job Monitoring MIB is intended to be implemented by an agent within
165 a printer or the first server closest to the printer, where the printer
166 is either directly connected to the server only or the printer does not
167 contain the job monitoring MIB agent. It is recommended that
168 implementations place the SNMP agent as close as possible to the
169 processing of the print job. This MIB applies to printers with and
170 without spooling capabilities. This MIB is designed to be compatible
171 with most current commonly-used job submission protocols. In most
172 environments that support high function job submission/job control
173 protocols, like ISO DPA[iso-dpa], those protocols would be used to
174 monitor and manage print jobs rather than using the Job Monitoring MIB.

175 The Job Monitoring MIB consists of a General Group, a Job Submission ID
176 Group, a Job Group, and an Attribute Group. Each group is a table.
177 All accessible objects are read-only. The General Group contains
178 general information that applies to all jobs in a job set. The Job
179 Submission ID table maps the job submission ID that the client uses to
180 identify a job to the jmJobIndex that the Job Monitoring Agent uses to
181 identify jobs in the Job and Attribute tables. The Job table contains
182 the MANDATORY integer job state and status objects. The Attribute
183 table consists of multiple entries per job that specify (1) job and
184 document identification and parameters, (2) requested resources, and
185 (3) consumed resources during and after job processing/printing. A
186 larger number of job attributes are defined as textual conventions that
187 an agent SHALL return if the server or device implements the
188 functionality so represented and the agent has access to the
189 information.

190 **1.1 Types of Information in the MIB**

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

194 User:

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

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

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

203 Provide error and diagnostic information for jobs that did not
204 successfully complete.

205 Operator:

206 Provide a presentation of the state of all the jobs in the print
207 system.

208 Provide the ability to identify the user that submitted the print
209 job.

210 Provide the ability to identify the resources required by each
211 job.

212 Provide the ability to define which physical printers are
213 candidates for the print job.

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

218 Capacity Planner:

219 Provide the ability to determine printer utilization as a
220 function of time.

221 Provide the ability to determine how long jobs wait before
222 starting to print.

223 Accountant:

224 Provide information to allow the creation of a record of
225 resources consumed and printer usage data for charging users or
226 groups for resources consumed.

227 Provide information to allow the prediction of consumable usage
228 and resource need.

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

235 1.2 Types of Job Monitoring Applications

236 The Job Monitoring MIB is designed for the following types of
237 monitoring applications:

- 238 1. Monitor a single job starting when the job is submitted and
239 ending a defined period after the job completes. The Job
240 Submission ID table provides the map to find the specific job
241 to be monitored.
- 242 2. Monitor all 'active' jobs in a queue, which this specification
243 generalizes to a "job set". End users may use such a program
244 when selecting a least busy printer, so the MIB is designed for
245 such a program to start up quickly and find the information
246 needed quickly without having to read all (completed) jobs in
247 order to find the active jobs. System operators may also use
248 such a program, in which case it would be running for a long
249 period of time and may also be interested in the jobs that have
250 completed. Finally such a program may be used to provide an
251 enhanced console and logging capability.
- 252 3. Collect resource usage for accounting or system utilization
253 purposes that copy the completed job statistics to an
254 accounting system. It is recognized that depending on
255 accounting programs to copy MIB data during the job-retention
256 period is somewhat unreliable, since the accounting program may
257 not be running (or may have crashed). Such a program is also
258 expected to keep a shadow copy of the entire Job Attribute
259 table including completed, canceled, and aborted jobs which the
260 program updates on each polling cycle. Such a program polls at
261 the rate of the persistence of the Attribute table. The design
262 is not optimized to help such an application determine which
263 jobs are completed, canceled, or aborted. Instead, the
264 application ~~SHALL-SHOULD~~ query each job that the application's
265 shadow copy shows was not complete, canceled, or aborted at the
266 previous poll cycle to see if it is now complete or canceled,
267 plus any new jobs that have been submitted.

268 The MIB provides a set of objects that represent a compatible subset of
269 job and document attributes of the ISO DPA standard[iso-dpa] and the
270 Internet Printing Protocol (IPP)[ipp-model], so that coherence is
271 maintained between these two protocols and the information presented to
272 end users and system operators by monitoring applications. However,
273 the job monitoring MIB is intended to be used with printers that
274 implement other job submitting and management protocols, such as IEEE
275 1284.1 (TIPSI)[tipsi], as well as with ones that do implement ISO DPA.

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

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

281 2 Terminology and Job Model

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

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

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

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

297 Attribute: A name, value-pair that specifies a job or document
298 instruction, a status, or a condition of a job or a document that has
299 been submitted to a server or device. A particular attribute NEED NOT
300 be present in each job instance. In other words, attributes are
301 present in a job instance only when there is a need to express the
302 value, either because (1) the client supplied a value in the job
303 submission protocol, (2) the document data contained an embedded
304 attribute, or (3) the server or device supplied a default value. An
305 agent ~~SHALL-MAY~~ represent an attribute as an entry (row) in the
306 Attribute table in this MIB in which entries are present only when
307 necessary. Attributes are identified in this MIB by an enum.

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

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

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

320 Document Instruction: An instruction specifying how to process the
321 document. Document instructions MAY be passed in the job submission
322 protocol separate from the actual document data, or MAY be embedded in
323 the document data or a combination, depending on the job submission
324 protocol and implementation.

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

327 Impression: For a print job, an impression is the passage of the
328 entire side of a sheet by the marker, whether or not any marks are made
329 and independent of the number of passes that the side makes past the
330 marker. Thus a four pass color process counts as a single impression,
331 as does highlight color. Impression counters count all kinds:
332 monochrome, highlight color, and full process color, while full color
333 counters only count full color impressions, and high light color
334 counters only count high light color impressions.

335 One-sided processing involves one impression per sheet. Two-sided
336 processing involves two impressions per sheet. If a two-sided document
337 has an odd number of pages, the last sheet still counts as two
338 impressions, if that sheet makes two passes through the marker or the
339 marker marks on both sides of a sheet in a single pass. Two-up
340 printing is the placement of two logical pages on one side of a sheet
341 and so is still a single impression. See "page" and "sheet".

342 NOTE - Since impressions include blank sides, it is suggested that
343 accounting application implementers consider charging for sheets,
344 rather than impressions, possibly using the value of the sides
345 attribute to select different charges for one-sided versus two-sided
346 printing, since some users may think that impressions don't include
347 blank sides.

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

351 Job: A unit of work whose results are expected together without
352 interjection of unrelated results. A job contains one or more
353 *documents*.

354 Job Accounting: The activity of a management application of accessing
355 the MIB and recording what happens to the job during and after the
356 processing of the job.

357 Job Instruction: An instruction specifying how, when, or where the job
358 is to be processed. Job instructions MAY be passed in the job
359 submission protocol or MAY be embedded in the document data or a
360 combination depending on the job submission protocol and
361 implementation.

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

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

370 Job Set: A group of jobs that are queued and scheduled together
371 according to a specified scheduling algorithm for a specified device or
372 set of devices. For implementations that embed the SNMP agent in the
373 device, the MIB job set normally represents *all* the jobs known to the
374 device, so that the implementation only implements a single job set.
375 If the SNMP agent is implemented in a server that controls one or more
376 devices, each MIB job set represents a job queue for (1) a specific
377 device or (2) set of devices, if the server uses a single queue to load
378 balance between several devices. Each job set is disjoint; no job
379 SHALL be represented in more than one MIB job set.

380 Monitor: Short for Job Monitoring Application.

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

384 Proxy: An agent that acts as a concentrator for one or more other
385 agents by accepting SNMP operations on the behalf of one or more other
386 agents, forwarding them on to those other agents, gathering responses
387 from those other agents and returning them to the original requesting
388 monitor.

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

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

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

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

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

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

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

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

408 Supervisor: A server that contains a control program that controls a
409 printer or other device. A supervisor is a client to the printer or
410 other device.

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

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

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

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

417 **1.12.1 System Configurations for the Job Monitoring MIB**

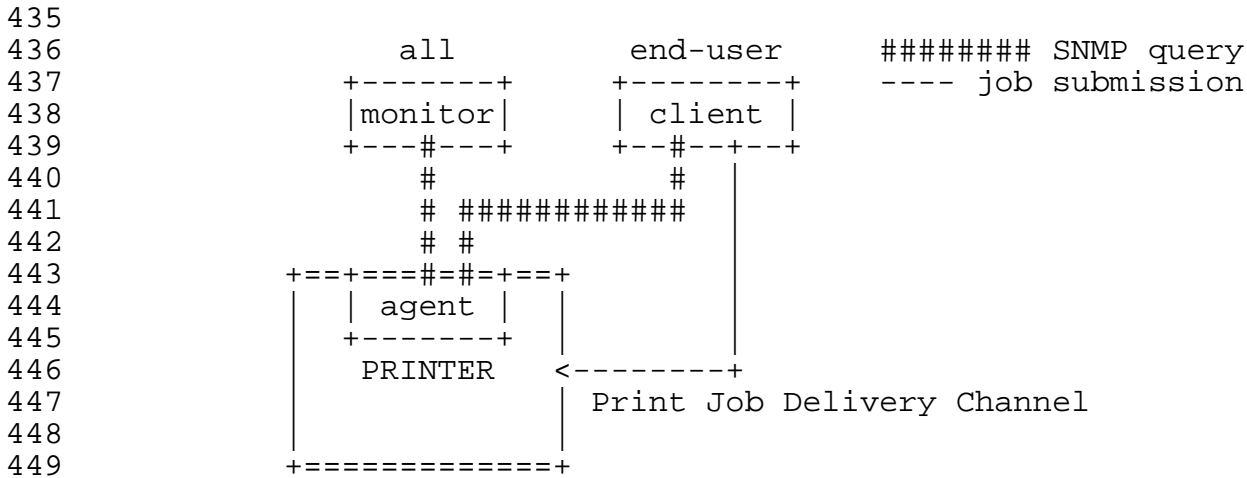
418 This section enumerates the three configurations in which the Job
419 Monitoring MIB is intended to be used. To simplify the pictures, the
420 *devices* are shown as *printers*. See section 1.1 entitled "Types of
421 Information in the MIB".

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

425 2.1.1 Configuration 1 - client-printer

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

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



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

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

- 453 1. Multiple clients MAY submit jobs to a printer.
- 454 2. Multiple clients MAY monitor a printer.
- 455 3. Multiple monitors MAY monitor a printer.
- 456 4. A client MAY submit jobs to multiple printers.
- 457 5. A monitor MAY monitor multiple printers.

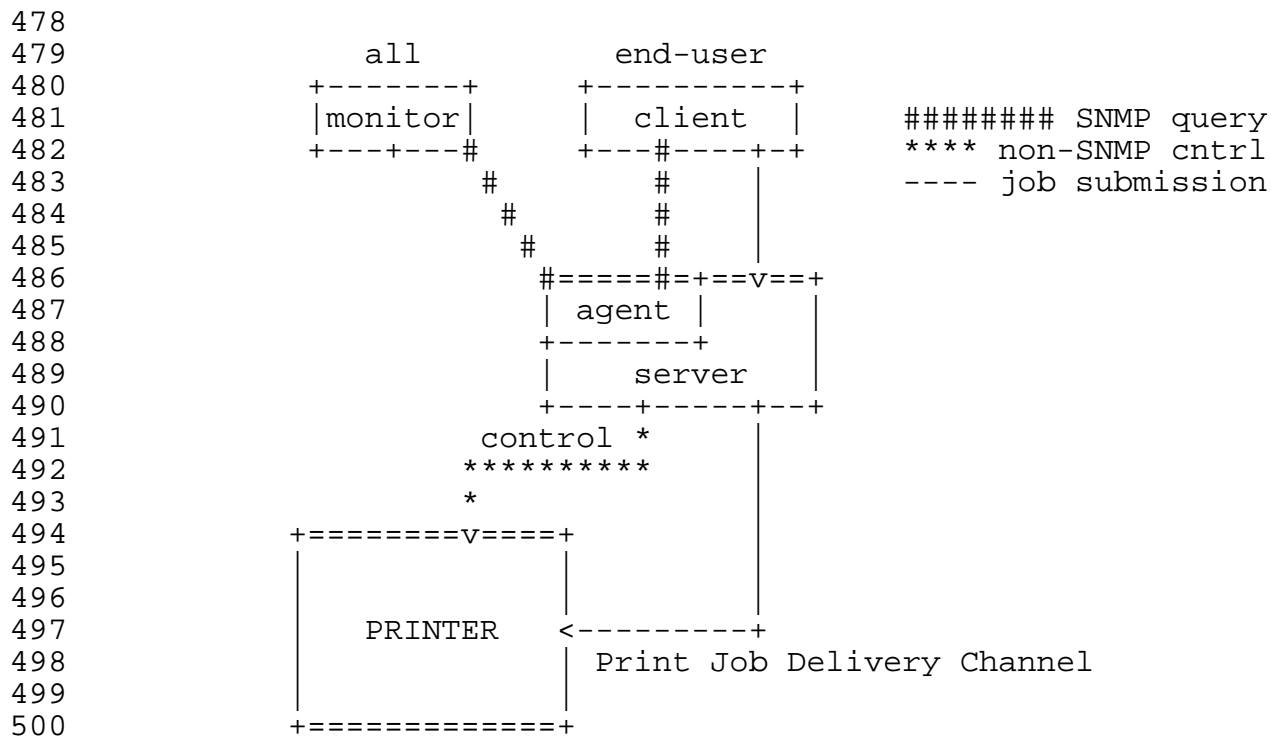
458 2.1.2 Configuration 2 - client-server-printer - agent in the server

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

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

465 A Job Monitoring MIB agent that is part of the server (or a front
466 for the server)

467 There is no SNMP Job Monitoring MIB agent in the printer in
468 configuration 2, at least that the client or monitor are aware. In
469 this configuration, the agent SHALL return the current values of the
470 objects in the Job Monitoring MIB both for jobs the server keeps and
471 jobs that the server has submitted to the printer. The Job Monitoring
472 MIB agent ~~SHALL~~ obtains the required information from the printer by a
473 method that is beyond the scope of this document. The agent in the
474 server SHALL keep the job in the Job Monitoring MIB in the server as
475 long as the job is in the printer, plus a defined time period after the
476 job enters the completed state in which accounting programs can copy
477 out the accounting data from the Job Monitoring MIB.



501 Figure 2-2 - Configuration 2 - client-server-printer - agent in the
 502 server

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

- 505 1. Multiple clients MAY submit jobs to a server.
- 506 2. Multiple clients MAY monitor a server.
- 507 3. Multiple monitors MAY monitor a server.
- 508 4. A client MAY submit jobs to multiple servers.
- 509 5. A monitor MAY monitor multiple servers.
- 510 6. Multiple servers MAY submit jobs to a printer.
- 511 7. Multiple servers MAY control a printer.

512 2.1.3 Configuration 3 - client-server-printer - client monitors printer 513 agent and server

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

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

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

572 3 Managed Object Usage

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

574 ~~1.13.1~~ **Conformance Considerations**

575 In order to achieve interoperability between job monitoring
576 applications and job monitoring agents, this specification includes the
577 conformance requirements for both monitoring applications and agents.

578 ~~1.1.13.1.1~~ Conformance Terminology

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

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

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

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

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

594 ~~1.1.23.1.2~~ Agent Conformance Requirements

595 A conforming agent:

- 596 1. SHALL implement *all* MANDATORY groups in this specification.
- 597 2. SHALL implement any attributes if (1) the server or device
598 supports the functionality represented by the attribute and (2)
599 the information is available to the agent.
- 600 3. SHOULD implement both forms of an attribute if it implements an
601 attribute that permits a choice of INTEGER and OCTET STRING
602 forms, since implementing both forms may help management
603 applications by giving them a choice of representations, since
604 the representation are equivalent. See the JmAttributeTypeTC
605 textual-convention.

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

608 ~~1.1.1.1~~ 3.1.2.1 MIB II System Group objects

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

612 ~~1.1.1.2~~ 3.1.2.2 MIB II Interface Group objects

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

616 ~~1.1.1.3~~ 3.1.2.3 Printer MIB objects

617 If the agent is providing access to a device that is a printer, the
618 agent SHALL implement all of the MANDATORY objects in the Printer
619 MIB[print-mib] and all the objects in other MIBs that conformance to
620 the Printer MIB requires, such as the Host Resources MIB[hr-mib]. If
621 the agent is providing access to a server that controls one or more
622 direct-connect or networked printers, the agent NEED NOT implement the
623 Printer MIB and NEED NOT implement the Host Resources MIB.

624 ~~1.1.3~~ 3.1.3 Job Monitoring Application Conformance Requirements

625 A conforming job monitoring application:

- 626 1. SHALL accept the full syntactic range for all objects in all
627 MANDATORY groups and all MANDATORY attributes that are required
628 to be implemented by an agent according to Section 3.1.2 and
629 SHALL either present them to the user or ignore them.
- 630 2. SHALL accept the full syntactic range for *all* attributes,
631 including enum and bit values specified in this specification
632 and additional ones that may be registered with the PWG and
633 SHALL either present them to the user or ignore them. In
634 particular, a conforming job monitoring application SHALL not
635 malfunction when receiving any standard or registered enum or
636 bit values. See Section 3.7 entitled "IANA and PWG
637 Registration Considerations".
- 638 3. SHALL NOT fail when operating with agents that materialize
639 attributes *after* the job has been submitted, as opposed to when
640 the job is submitted.
- 641 4. SHALL, if it supports a time attribute, accept either form of
642 the time attribute, since agents are free to implement either
643 time form.

644 3.2 The Job Tables and the Oldest Active and Newest Active Indexes

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

- 647 1. jmGeneralJobSetIndex - which job set
- 648 2. jmJobIndex - which job in the job set

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

- 652 1. jmGeneralOldestActiveJobIndex - the index of the active job
653 that has been in the tables the longest.
- 654 2. jmGeneralNewestActiveJobIndex - the index of the active job
655 that has been most recently added to the tables.

656 The agent SHALL assign the next incremental value of jmJobIndex to the
657 job, when a new job is accepted by the server or device to which the
658 agent is providing access. If the incremented value of jmJobIndex
659 would exceed the implementation-defined maximum value for jmJobIndex,
660 the agent SHALL 'wrap' back to 1. An agent uses the resulting value of
661 jmJobIndex for storing information in the jmJobTable and the
662 jmAttributeTable about the job.

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

668 It is recommended that agents that are providing access to
669 servers/devices that already allocate job-identifiers for jobs as
670 integers use the same integer value for the jmJobIndex. Then
671 management applications using this MIB and applications using other
672 protocols will see the same job identifiers for the same jobs. Agents
673 providing access to systems that contain jobs with a job identifier of
674 0 SHALL map the job identifier value 0 to a jmJobIndex value that is
675 one higher than the highest job identifier value that any job can have
676 on that system. Then only job 0 will have a different job-identifier
677 value than the job's jmJobIndex value.

678 NOTE - If a server or device accepts jobs using multiple job submission
679 protocols, it may be difficult for the agent to meet the recommendation
680 to use the job-identifier values that the server or device assigns as
681 the jmJobIndex value, unless the server/device assigns job-identifiers
682 for each of its job submission protocols from the same job-identifier
683 number space.

684 Each time a new job is accepted by the server or device that the agent
685 is providing access to AND that job is to be 'active' (pending,
686 processing, or processingStopped, but not pendingHeld), the agent SHALL
687 copy the value of the job's jmJobIndex to the
688 jmGeneralNewestActiveJobIndex object. If the new job is to be
689 'inactive' (pendingHeld state), the agent SHALL not change the value of
690 jmGeneralNewestActiveJobIndex object (though the agent SHALL assign the
691 next incremental jmJobIndex value to the job).

692 When a job transitions from one of the 'active' job states (pending,
693 processing, processingStopped) to one of the 'inactive' job states
694 (pendingHeld, completed, canceled, or aborted), with a jmJobIndex value
695 that matches the jmGeneralOldestActiveJobIndex object, the agent SHALL
696 advance (or wrap) the value to the next oldest 'active' job, if any.
697 See the JmJobStateTC textual-convention for a definition of the job
698 states.

699 Whenever a job transitions from one of the 'inactive' job states to one
700 of the 'active' job states (from pendingHeld to pending or processing),
701 the agent SHALL update the value of either the
702 jmGeneralOldestActiveJobIndex or the jmGeneralNewestActiveJobIndex
703 objects, or both, if the job's jmJobIndex value is outside the range
704 between jmGeneralOldestActiveJobIndex and
705 jmGeneralNewestActiveJobIndex.

706 When all jobs become 'inactive', i.e., enter the pendingHeld,
707 completed, canceled, or aborted states, the agent SHALL set the value
708 of both the jmGeneralOldestActiveJobIndex and
709 jmGeneralNewestActiveJobIndex objects to 0.

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

715 If an application detects that the jmGeneralNewestActiveJobIndex is
716 smaller than jmGeneralOldestActiveJobIndex, the job index has wrapped.
717 In this case, the application SHALL reset the index to 1 when the end
718 of the table is reached and continue the GetNext operations to find the
719 rest of the active jobs.

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

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

727 3.3 The Attribute Mechanism

728 Attributes are similar to information objects, except that attributes
729 are identified by an enum, instead of an OID, so that attributes may be
730 registered without requiring a new MIB. Also an implementation that
731 does not have the functionality represented by the attribute can omit
732 the attribute entirely, rather than having to return a distinguished
733 value. The agent is free to materialize an attribute in the
734 jmAttributeTable as soon as the agent is aware of the value of the
735 attribute.

736 The agent materializes job attributes in a four-indexed
737 jmAttributeTable:

- 738 1. jmGeneralJobSetIndex - which job set
- 739 2. jmJobIndex - which job in the job set
- 740 3. jmAttributeTypeIndex - which attribute
- 741 4. jmAttributeInstanceIndex - which attribute instance for those
742 attributes that can have multiple values per job.

743 Some attributes represent information about a job, such as a file-name,
744 a document-name, a submission-time or a completion time. Other
745 attributes represent resources required, e.g., a medium or a colorant,
746 etc. to process the job before the job starts processing OR to indicate
747 the amount of the resource consumed during and after processing, e.g.,
748 pages completed or impressions completed. If both a required and a
749 consumed value of a resource is needed, this specification assigns two
750 separate attribute enums in the textual convention.

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

758 3.3.1 Conformance of Attribute Implementation

759 An agent SHALL implement any attribute if (1) the server or device
760 supports the functionality represented by the attribute and (2) the
761 information is available to the agent. The agent MAY create the
762 attribute row in the jmAttributeTable when the information is available
763 or MAY create the row earlier with the designated 'unknown' value
764 appropriate for that attribute. See next section.

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

768 3.3.2 Useful, 'Unknown', and 'Other' Values for Objects and Attributes

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

773 SNMP requires that if an object cannot be implemented because its
774 values cannot be accessed, then a compliant agent SHALL return an SNMP
775 error in SNMPv1 or an exception value in SNMPv2. However, this MIB has
776 been designed so that 'all' objects can and SHALL be implemented by an
777 agent, so that neither the SNMPv1 error nor the SNMPv2 exception value
778 SHALL be generated by the agent. This MIB has also been designed so
779 that when an agent materializes an attribute, the agent SHALL
780 materialize a row consisting of both the jmAttributeValueAsInteger and
781 jmAttributeValueAsOctets objects.

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

789 Since each attribute is represented by a row consisting of both the
790 jmAttributeValueAsInteger and jmAttributeValueAsOctets MANDATORY
791 objects, SNMP requires that the agent SHALL always create an attribute
792 row with both objects specified. However, for most attributes the
793 agent SHALL return a "useful" value for one of the objects and SHALL
794 return the 'other' value for the other object. For integer only
795 attributes, the agent SHALL always return a zero-length string value
796 for the jmAttributeValueAsOctets object. For octet string only
797 attributes, the agent SHALL always return a '-1' value for the
798 jmAttributeValueAsInteger object.

799 3.3.3 Index Value Attributes

800 A number of attributes are indexes in other tables. Such attribute
801 names end with the word 'Index'. If the agent has not (yet) assigned
802 an index value for a particular index attribute for a job, the agent
803 SHALL either: (1) return the value 0 or (2) not add this attribute to
804 the jmAttributeTable until the index value is assigned. In the
805 interests of brevity, the semantics for 0 is specified once here and is
806 not repeated for each index attribute specification and a DEFVAL of 0
807 is indicated implied, even though the DEFVAL for
808 jmAttributeValueAsInteger is -2.

809 3.3.4 Data Sub-types and Attribute Naming Conventions

810 Many attributes are sub-typed to give a more specific data type than
811 Integer32 or OCTET STRING. The data sub-type of each attribute is
812 indicated on the first line(s) of the description. Some attributes
813 have several different data sub-type representations. When an
814 attribute has both an Integer32 data sub-type and an OCTET STRING data
815 sub-type, the attribute can be represented in a single row in the
816 jmAttributeTable. In this case, the data sub-type name is not included
817 as the last part of the name of the attribute, e.g., documentFormat(38)
818 which is both an enum and/or a name. When the data sub-types cannot be
819 represented by a single row in the jmAttributeTable, each such
820 representation is considered a separate attribute and is assigned a
821 separate name and enum value. For these attributes, the name of the
822 data sub-type is the last part of the name of the attribute: Name,
823 Index, DateAndTime, TimeStamp, etc. For example,
824 documentFormatIndex(37) is an index.

825 NOTE: The Table of Contents also lists the data sub-type and/or data
826 sub-types of each attribute, using the textual-convention name when
827 such is defined. The following abbreviations are used in the Table of
828 Contents as shown:
829

'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.

830 3.3.5 Single-Value (Row) Versus Multi-Value (MULTI-ROW) Attributes

831 Most attributes ~~SHALL~~ have only one row per job. However, a few
832 attributes can have multiple values per job or even per document, where
833 each value is a separate row in the jmAttributeTable. Unless indicated
834 with 'MULTI-ROW:' in the JmAttributeTypeTC description, an agent SHALL
835 ensure that each attribute occurs only once in the jmAttributeTable for
836 a job. Most of the 'MULTI-ROW' attributes do not allow duplicate
837 values, i.e., the agent SHALL ensure that each value occurs only once
838 for a job. Only if the specification of the 'MULTI-ROW' attribute also
839 says "There is no restriction on the same xxx occurring in multiple
840 rows" can the agent allow duplicate values to occur for the job.

841 NOTE - Duplicates are allowed for 'extensive' 'MULTI-ROW' attributes,
842 such as fileName(34) or documentName(35) which are specified to be

843 'per-document' attributes, but are *not* allowed for 'intensive' 'MULTI-
844 ROW' attributes, such as mediumConsumed(171) and documentFormat(38)
845 which are specified to be 'per-job' attributes.

846 3.3.6 Requested Objects and Attributes

847 A number of objects and attributes record requirements for the job.
848 Such object and attribute names end with the word 'Requested'. In the
849 interests of brevity, the phrase 'requested' ~~SHALL~~ means: (1) requested
850 by the client (or intervening server) in the job submission protocol
851 and ~~MAY~~ may also mean (2) embedded in the submitted document data,
852 and/or (3) defaulted by the recipient device or server with the same
853 semantics as if the requester had supplied, depending on
854 implementation. Also if a value is supplied by the job submission
855 client, and the server/device determines a better value, through
856 processing or other means, the agent MAY return that better value for
857 such object and attribute.

858 3.3.7 Consumption Attributes

859 A number of objects and attributes record consumption. Such attribute
860 names end with the word 'Completed' or 'Consumed'. If the job has not
861 yet consumed what that resource is metering, the agent either: (1)
862 SHALL return the value 0 or (2) SHALL *not* add this attribute to the
863 jmAttributeTable until the consumption begins. In the interests of
864 brevity, the semantics for 0 is specified once here and is *not* repeated
865 for each consumption attribute specification and a DEFVAL of 0 is
866 ~~indicated~~ implied, even though the DEFVAL for jmAttributeValueAsInteger
867 is -2.

868 3.3.8 Attribute Specifications

869 This section specifies the job attributes.

870 In the following definitions of the attributes, each description
871 indicates whether the useful value of the attribute SHALL be
872 represented using the jmAttributeValueAsInteger or the
873 jmAttributeValueAsOctets objects by the initial tag: 'INTEGER:' or
874 'OCTETS:', respectively.

875 Some attributes allow the agent implementer a choice of useful values
876 of either an integer, an octets representation, or both, depending on
877 implementation. These attributes are indicated with 'INTEGER:' AND/OR
878 'OCTETS:' tags.

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

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

886 Values in the range 2**30 to 2**31-1 are reserved for private or
 887 experimental usage. This range corresponds to the same range reserved
 888 in IPP. Implementers are warned that use of such values may conflict
 889 with other implementations. Implementers are encouraged to request
 890 registration of enum values following the procedures in Section 3.7.1.

891 NOTE: No attribute name exceeds 31 characters.

892 The standard attribute types are:

893	jmAttributeTypeIndex	Datatype
894	-----	-----
896	other(1),	Integer32 (-2..2147483647)
897		AND/OR
898		OCTET STRING(SIZE(0..63))
899		
900	INTEGER: and/or	OCTETS: An attribute that is not in the
901	list and/or that has not been	approved and registered with
902	the PWG.	

903 ++++++
904 + Job State attributes
905 +
906 + The following attributes specify the state of a job.
907 ++++++

908
909 jobStateReasons2(3), JmJobStateReasons2TC
910 INTEGER: Additional information about the job's current
911 state that augments the jmJobState object. See the
912 description under the JmJobStateReasons1TC textual-
913 convention.
914

915 jobStateReasons3(4), JmJobStateReasons3TC
916 INTEGER: Additional information about the job's current
917 state that augments the jmJobState object. See the
918 description under JmJobStateReasons1TC textual-convention.
919

920 jobStateReasons4(5), JmJobStateReasons4TC
921 INTEGER: Additional information about the job's current
922 state that augments the jmJobState object. See the
923 description under JmJobStateReasons1TC textual-convention.
924

925 processingMessage(6), JmUTF8StringTC (SIZE(0..63))
926 OCTETS: MULTI-ROW: A coded character set message that is
927 generated by the server or device during the processing of
928 the job as a simple form of processing log to show progress
929 and any problems. The natural language of each value is
930 specified by the corresponding
931 processingMessageNaturalLangTag(7) value.
932

933 NOTE - This attribute is intended for such conditions as
934 interpreter messages, rather than being the printable form
935 of the jmJobState and jmJobStateReasons1 objects and
936 jobStateReasons2, jobStateReasons3, and jobStateReasons4
937 attributes. In order to produce a localized printable form
938 of these job state objects/attribute, a management
939 application SHOULD produce a message from their enum and
940 bit values.
941

942 NOTE - There is no job description attribute in IPP/1.0
943 that corresponds to this attribute and this attribute does
944 not correspond to the IPP/1.0 'job-state-message' job
945 description attribute, which is just a printable form of
946 the IPP 'job-state' and 'job-state-reasons' job attributes.
947

948 There is no restriction for the same message occurring in
949 multiple rows.

950
951 processingMessageNaturalLangTag(7), OCTET STRING(SIZE(0..63))
952 OCTETS: MULTI-ROW: The natural language of the
953 corresponding processingMessage(6) attribute value. See
954 section 3.6.1, entitled 'Text generated by the server or
955 device'.
956
957 If the agent does not know the natural language of the job
958 processing message, the agent SHALL either (1) return a
959 zero length string value for the
960 processingMessageNaturalLangTag(7) attribute or (2) not
961 return the processingMessageNaturalLangTag(7) attribute for
962 the job.
963
964 There is no restriction for the same tag occurring in
965 multiple rows, since when this attribute is implemented, it
966 SHOULD have a value row for each corresponding
967 processingMessage(6) attribute value row.
968
969 jobCodedCharSet(8), CodedCharSet
970 INTEGER: The MIBenum identifier of the coded character set
971 that the agent is using to represent coded character set
972 objects and attributes of type 'JmJobStringTC'. These
973 coded character set objects and attributes are either: (1)
974 supplied by the job submitting client or (2) defaulted by
975 the server or device when omitted by the job submitting
976 client. The agent SHALL represent these objects and
977 attributes in the MIB either (1) in the coded character set
978 as they were submitted or (2) MAY convert the coded
979 character set to another coded character set or encoding
980 scheme as identified by the jobCodedCharSet(8) attribute.
981 See section 3.6.2, entitled 'Text supplied by the job
982 submitter'.
983
984 These MIBenum values are assigned by IANA [IANA-charsets]
985 when the coded character sets are registered. The coded
986 character set SHALL be one of the ones registered with IANA
987 [IANA] and the enum value uses the CodedCharSet textual-
988 convention from the Printer MIB. See the JmJobStringTC
989 textual-convention.
990
991 If the agent does not know what coded character set was
992 used by the job submitting client, the agent SHALL either
993 (1) return the 'unknown(2)' value for the
994 jobCodedCharSet(8) attribute or (2) not return the
995 jobCodedCharSet(8) attribute for the job.
996
997
998 jobNaturalLanguageTag(9), OCTET STRING(SIZE(0..63))
999 OCTETS: The natural language of the job attributes supplied
1000 by the job submitter or defaulted by the server or device
1001 for the job, i.e., all objects and attributes represented

1002 by the 'JmJobStringTC' textual-convention, such as jobName,
1003 mediumRequested, etc. See Section 3.6.2, entitled 'Text
1004 supplied by the job submitter'.
1005

1006 If the agent does not know what natural language was used
1007 by the job submitting client, the agent SHALL either (1)
1008 return a zero length string value for the
1009 jobNaturalLanguageTag(9) attribute or (2) not return
1010 jobNaturalLanguageTag(9) attribute for the job.
1011

1012 ++++++
1013 + Job Identification attributes
1014 +
1015 + The following attributes help an end user, a system
1016 + operator, or an accounting program identify a job.
1017 ++++++

1018

1019 jobURI(20), OCTET STRING(SIZE(0..63))
1020 OCTETS: MULTI-ROW: The job's Universal Resource
1021 Identifier (URI) [RFC-1738]. See IPP [ipp-model] for
1022 example usage.
1023

1024 NOTE - The agent may be able to generate this value on each
1025 SNMP Get operation from smaller values, rather than having
1026 to store the entire URI.
1027

1028 If the URI exceeds 63 octets, the agent SHALL use multiple
1029 values, with the next 63 octets coming in the second value,
1030 etc.
1031

1032 NOTE - IPP [ipp-model] has a 1023-octet maximum length for
1033 a URI, though the URI standard itself and HTTP/1.1 specify
1034 no maximum length.
1035

1036 jobAccountName(21), OCTET STRING(SIZE(0..63))
1037 OCTETS: Arbitrary binary information which MAY be coded
1038 character set data or encrypted data supplied by the
1039 submitting user for use by accounting services to allocate
1040 or categorize charges for services provided, such as a
1041 customer account name or number.
1042

1043 NOTE: This attribute NEED NOT be printable characters.
1044

1045 serverAssignedJobName(22), JmJobStringTC (SIZE(0..63))
1046 OCTETS: Configuration 3 only: The human readable string
1047 name, number, or ID of the job as assigned by the server
1048 that submitted the job to the device that the agent is
1049 providing access to with this MIB.
1050
1051 NOTE - This attribute is intended for enabling a user to
1052 find his/her job that a server submitted to a device when
1053 either the client does not support the jmJobSubmissionID or
1054 the server does not pass the jmJobSubmissionID through to
1055 the device.
1056
1057 jobName(23), JmJobStringTC (SIZE(0..63))
1058 OCTETS: The human readable string name of the job as
1059 assigned by the submitting user to help the user
1060 distinguish between his/her various jobs. This name does
1061 not need to be unique.
1062
1063 This attribute is intended for enabling a user or the
1064 user's application to convey a job name that MAY be printed
1065 on a start sheet, returned in a query result, or used in
1066 notification or logging messages.
1067
1068 In order to assist users to find their jobs for job
1069 submission protocols that don't supply a jmJobSubmissionID,
1070 the agent SHOULD maintain the jobName attribute for the
1071 time specified by the jmGeneralJobPersistence object,
1072 rather than the (shorter) jmGeneralAttributePersistence
1073 object.
1074
1075 If this attribute is not specified when the job is
1076 submitted, no job name is assumed, but implementation
1077 specific defaults are allowed, such as the value of the
1078 documentName attribute of the first document in the job or
1079 the fileName attribute of the first document in the job.
1080
1081 The jobName attribute is distinguished from the jobComment
1082 attribute, in that the jobName attribute is intended to
1083 permit the submitting user to distinguish between different
1084 jobs that he/she has submitted. The jobComment attribute
1085 is intended to be free form additional information that a
1086 user might wish to use to communicate with himself/herself,
1087 such as a reminder of what to do with the results or to
1088 indicate a different set of input parameters were tried in
1089 several different job submissions.
1090

1091 jobServiceTypes(24), JmJobServiceTypesTC
1092 INTEGER: Specifies the type(s) of service to which the job
1093 has been submitted (print, fax, scan, etc.). The service
1094 type is bit encoded with each job service type so that more
1095 general and arbitrary services can be created, such as
1096 services with more than one destination type, or ones with
1097 only a source or only a destination. For example, a job
1098 service might scan, faxOut, and print a single job. In
1099 this case, three bits would be set in the jobServiceTypes
1100 attribute, corresponding to the hexadecimal values: 0x8 +
1101 0x20 + 0x4, respectively, yielding: 0x2C.
1102
1103 Whether this attribute is set from a job attribute supplied
1104 by the job submission client or is set by the recipient job
1105 submission server or device depends on the job submission
1106 protocol. This attribute SHALL be implemented if the
1107 server or device has other types in addition to or instead
1108 of printing.
1109
1110 One of the purposes of this attribute is to permit a
1111 requester to filter out jobs that are not of interest. For
1112 example, a printer operator may only be interested in jobs
1113 that include printing.
1114
1115 jobSourceChannelIndex(25), Integer32 (0..2147483647)
1116 INTEGER: The index of the row in the associated Printer
1117 MIB[print-mib] of the channel which is the source of the
1118 print job.
1119
1120 jobSourcePlatformType(26), JmJobSourcePlatformTypeTC
1121 INTEGER: The source platform type of the immediate
1122 upstream submitter that submitted the job to the server
1123 (configuration 2) or device (configuration 1 and 3) to
1124 which the agent is providing access. For configuration 1,
1125 this is the type of the client that submitted the job to
1126 the device; for configuration 2, this is the type of the
1127 client that submitted the job to the server; and for
1128 configuration 3, this is the type of the server that
1129 submitted the job to the device.
1130
1131 submittingServerName(27), JmJobStringTC (SIZE(0..63))
1132 OCTETS: For configuration 3 only: The administrative name
1133 of the server that submitted the job to the device.
1134
1135 submittingApplicationName(28), JmJobStringTC (SIZE(0..63))
1136 OCTETS: The name of the client application (not the server
1137 in configuration 3) that submitted the job to the server or
1138 device.
1139

1140 jobOriginatingHost(29), JmJobStringTC (SIZE(0..63))
1141 OCTETS: The name of the client host (not the server host
1142 name in configuration 3) that submitted the job to the
1143 server or device.
1144
1145 deviceNameRequested(30), JmJobStringTC (SIZE(0..63))
1146 OCTETS: The administratively defined coded character set
1147 name of the target device requested by the submitting user.
1148 For configuration 1, its value corresponds to the Printer
1149 MIB[print-mib]: prtGeneralPrinterName object. For
1150 configuration 2 and 3, its value is the name of the logical
1151 or physical device that the user supplied to indicate to
1152 the server on which device(s) they wanted the job to be
1153 processed.
1154
1155 queueNameRequested(31), JmJobStringTC (SIZE(0..63))
1156 OCTETS: The administratively defined coded character set
1157 name of the target queue requested by the submitting user.
1158 For configuration 1, its value corresponds to the queue in
1159 the device for which the agent is providing access. For
1160 configuration 2 and 3, its value is the name of the queue
1161 that the user supplied to indicate to the server on which
1162 device(s) they wanted the job to be processed.
1163
1164 NOTE - typically an implementation SHOULD support either
1165 the deviceNameRequested or queueNameRequested attribute,
1166 but not both.
1167
1168 physicalDevice(32), hrDeviceIndex
1169 AND/OR
1170 JmUTF8StringTC (SIZE(0..63))
1171 INTEGER: MULTI-ROW: The index of the physical device MIB
1172 instance requested/used, such as the Printer MIB[print-
1173 mib]. This value is an hrDeviceIndex value. See the Host
1174 Resources MIB[hr-mib].
1175
1176 AND/OR
1177
1178 OCTETS: MULTI-ROW: The name of the physical device to
1179 which the job is assigned.
1180
1181 numberOfDocuments(33), Integer32 (-2..2147483647)
1182 INTEGER: The number of documents in this job.
1183
1184 The agent SHOULD return this attribute if the job has more
1185 than one document.
1186


```

1228     documentFormat(38),                               PrtInterpreterLangFamilyTC
1229                                                     AND/OR
1230                                                     OCTET STRING(SIZE(0..63))
1231     INTEGER: MULTI-ROW: The interpreter language family
1232     corresponding to the Printer MIB[print-mib]
1233     prtInterpreterLangFamily object, that this job
1234     requires/uses. A document or a job MAY use more than one
1235     PDL or control language.
1236
1237     AND/OR
1238
1239     OCTETS: MULTI-ROW: The document format registered as a
1240     media type[iana-media-types], i.e., the name of the MIME
1241     content-type/subtype. Examples: 'application/postscript',
1242     'application/vnd.hp-PCL', 'application/pdf', 'text/plain'
1243     (US-ASCII SHALL be assumed), 'text/plain; charset=iso-8859-
1244     1', and 'application/octet-stream'. The IPP 'document-
1245     format' job attribute uses these same values with the same
1246     semantics. See the IPP [ipp-model] 'mimeMediaType'
1247     attribute syntax and the document-format attribute for
1248     further examples and explanation.
1249
1250     ++++++
1251     + Job Parameter attributes
1252     +
1253     + The following attributes represent input parameters
1254     + supplied by the submitting client in the job submission
1255     + protocol.
1256     ++++++
1257
1258     jobPriority(50),                                     Integer32 (-2..100)
1259     INTEGER: The priority for scheduling the job. It is used
1260     by servers and devices that employ a priority-based
1261     scheduling algorithm.
1262
1263     A higher value specifies a higher priority. The value 1 is
1264     defined to indicate the lowest possible priority (a job
1265     which a priority-based scheduling algorithm SHALL pass over
1266     in favor of higher priority jobs). The value 100 is
1267     defined to indicate the highest possible priority.
1268     Priority is expected to be evenly or 'normally' distributed
1269     across this range. The mapping of vendor-defined priority
1270     over this range is implementation-specific. -2 indicates
1271     unknown.
1272

```


1273 jobProcessAfterDateAndTime(51), DateAndTime (SNMPv2-TC)
1274 OCTETS: The calendar date and time of day after which the
1275 job SHALL become a candidate to be scheduled for
1276 processing. If the value of this attribute is in the
1277 future, the server SHALL set the value of the job's
1278 jmJobState object to pendingHeld and add the
1279 jobProcessAfterSpecified bit value to the job's
1280 jmJobStateReasons1 object. When the specified date and
1281 time arrives, the server SHALL remove the
1282 jobProcessAfterSpecified bit value from the job's
1283 jmJobStateReasons1 object and, if no other reasons remain,
1284 SHALL change the job's jmJobState object to pending.
1285
1286 jobHold(52), JmBooleanTC
1287 INTEGER: If the value is 'true(4)', a client has
1288 explicitly specified that the job is to be held until
1289 explicitly released. Until the job is explicitly released
1290 by a client, the job SHALL be in the pendingHeld state with
1291 the jobHoldSpecified value in the jmJobStateReasons1
1292 attribute.
1293
1294 jobHoldUntil(53), JmJobStringTC (SIZE(0..63))
1295 OCTETS: The named time period during which the job SHALL
1296 become a candidate for processing, such as 'evening',
1297 'night', 'weekend', 'second-shift', 'third-shift', etc.,
1298 (supported values configured as defined by the system
1299 administrator). See IPP [ipp-model] for the standard
1300 keyword values. Until that time period arrives, the job
1301 SHALL be in the pendingHeld state with the
1302 jobHoldUntilSpecified value in the jmJobStateReasons1
1303 object. The value 'no-hold' SHALL indicate explicitly that
1304 no time period has been specified; the absence of this
1305 attribute SHALL indicate implicitly that no time period has
1306 been specified.
1307
1308 outputBin(54), Integer32 (0..2147483647)
1309 AND/OR
1310 JmJobStringTC (SIZE(0..63))
1311 INTEGER: MULTI-ROW: The output subunit index in the
1312 Printer MIB[print-mib]
1313
1314 AND/OR
1315
1316 OCTETS: MULTI-ROW: the name or number (represented as
1317 ASCII digits) of the output bin to which all or part of the
1318 job is placed in.
1319

```
1320 sides(55), Integer32 (-2..2)
1321     INTEGER: MULTI-ROW: The number of sides, '1' or '2', that
1322     any document in this job requires/used.
1323
1324 finishing(56), JmFinishingTC
1325     INTEGER: MULTI-ROW: Type of finishing that any document
1326     in this job requires/used.
1327
1328
1329 ++++++
1330 + Image Quality attributes (requested and consumed)
1331 +
1332 + For devices that can vary the image quality.
1333 ++++++
1334
1335 printQualityRequested(70), JmPrintQualityTC
1336     INTEGER: MULTI-ROW: The print quality selection requested
1337     for a document in the job for printers that allow quality
1338     differentiation.
1339
1340 printQualityUsed(71), JmPrintQualityTC
1341     INTEGER: MULTI-ROW: The print quality selection actually
1342     used by a document in the job for printers that allow
1343     quality differentiation.
1344
1345 printerResolutionRequested(72), JmPrinterResolutionTC
1346     OCTETS: MULTI-ROW: The printer resolution requested for a
1347     document in the job for printers that support resolution
1348     selection.
1349
1350 printerResolutionUsed(73), JmPrinterResolutionTC
1351     OCTETS: MULTI-ROW: The printer resolution actually used
1352     by a document in the job for printers that support
1353     resolution selection.
1354
1355 tonerEcomonyRequested(74), JmTonerEcomonyTC
1356     INTEGER: MULTI-ROW: The toner economy selection requested
1357     for documents in the job for printers that allow toner
1358     economy differentiation.
1359
1360 tonerEcomonyUsed(75), JmTonerEcomonyTC
1361     INTEGER: MULTI-ROW: The toner economy selection actually
1362     used by documents in the job for printers that allow toner
1363     economy differentiation.
1364
1365 tonerDensityRequested(76) Integer32 (-2..100)
1366     INTEGER: MULTI-ROW: The toner density requested for a
1367     document in this job for devices that can vary toner
1368     density levels. Level 1 is the lowest density and level
1369     100 is the highest density level. Devices with a smaller
1370     range, SHALL map the 1-100 range evenly onto the
1371     implemented range.
```

1372
1373 tonerDensityUsed(77), Integer32 (-2..100)
1374 INTEGER: MULTI-ROW: The toner density used by documents
1375 in this job for devices that can vary toner density levels.
1376 Level 1 is the lowest density and level 100 is the highest
1377 density level. Devices with a smaller range, SHALL map the
1378 1-100 range evenly onto the implemented range.
1379
1380 ++++++
1381 + Job Progress attributes (requested and consumed)
1382 +
1383 + Pairs of these attributes can be used by monitoring
1384 + applications to show an indication of relative progress
1385 + to users. See section 3.4, entitled '**Monitoring Job**
1386 **Progress**'.
1387 ++++++
1388
1389 jobCopiesRequested(90), Integer32 (-2..2147483647)
1390 INTEGER: The number of copies of the entire job that are
1391 to be produced.
1392
1393 jobCopiesCompleted(91), Integer32 (-2..2147483647)
1394 INTEGER: The number of copies of the entire job that have
1395 been completed so far.
1396
1397 documentCopiesRequested(92), Integer32 (-2..2147483647)
1398 INTEGER: The total count of the number of document copies
1399 requested for the job as a whole. If there are documents
1400 A, B, and C, and document B is specified to produce 4
1401 copies, the number of document copies requested is 6 for
1402 the job.
1403
1404 This attribute SHALL be used only when a job has multiple
1405 documents. The jobCopiesRequested attribute SHALL be used
1406 when the job has only one document.
1407
1408 documentCopiesCompleted(93), Integer32 (-2..2147483647)
1409 INTEGER: The total count of the number of document copies
1410 completed so far for the job as a whole. If there are
1411 documents A, B, and C, and document B is specified to
1412 produce 4 copies, the number of document copies starts a 0
1413 and runs up to 6 for the job as the job processes.
1414
1415 This attribute SHALL be used only when a job has multiple
1416 documents. The jobCopiesCompleted attribute SHALL be used
1417 when the job has only one document.
1418

1419 jobKOctetsTransferred(94), Integer32 (-2..2147483647)
1420 INTEGER: The number of K (1024) octets transferred to the
1421 server or device to which the agent is providing access.
1422 This count is independent of the number of copies of the
1423 job or documents that will be produced, but it is only a
1424 measure of the number of bytes transferred to the server or
1425 device.
1426
1427 The agent SHALL round the actual number of octets
1428 transferred up to the next higher K. Thus 0 octets SHALL
1429 be represented as '0', 1-1024 octets SHALL BE represented
1430 as '1', 1025-2048 SHALL be '2', etc. When the job
1431 completes, the values of the jmJobKOctetsPerCopyRequested
1432 object and the jobKOctetsTransferred attribute SHALL be
1433 equal.
1434
1435 NOTE - The jobKOctetsTransferred can be used with the
1436 jmJobKOctetsPerCopyRequested object in order to produce a
1437 relative indication of the progress of the job for agents
1438 that do not implement the jmJobKOctetsProcessed object.
1439
1440 sheetCompletedCopyNumber(95), Integer32 (-2..2147483647)
1441 INTEGER: The number of the copy being stacked for the
1442 current document. This number starts at 0, is set to 1
1443 when the first sheet of the first copy for each document is
1444 being stacked and is equal to n where n is the nth sheet
1445 stacked in the current document copy. See section 3.4 ,
1446 entitled 'Monitoring Job Progress'.
1447
1448 sheetCompletedDocumentNumber(96), Integer32 (-2..2147483647)
1449 INTEGER: The ordinal number of the document in the job
1450 that is currently being stacked. This number starts at 0,
1451 increments to 1 when the first sheet of the first document
1452 in the job is being stacked, and is equal to n where n is
1453 the nth document in the job, starting with 1.
1454
1455 Implementations that only support one document jobs SHOULD
1456 NOT implement this attribute.
1457
1458 jobCollationType(97), JmJobCollationTypeTC
1459 INTEGER: The type of job collation. See also Section 3.4,
1460 entitled 'Monitoring Job Progress'.
1461
1462 +++++
1463 + Impression attributes
1464 +
1465 + See the definition of the terms 'impression', 'sheet',
1466 + and 'page' in Section 2.
1467 +
1468 + See also jmJobImpressionsPerCopyRequested and
1469 + jmJobImpressionsCompleted objects in the jmJobTable.
1470 +++++

1471
1472 impressionsSpooled(110), Integer32 (-2..2147483647)
1473 INTEGER: The number of impressions spooled to the server
1474 or device for the job so far.
1475
1476 impressionsSentToDevice(111), Integer32 (-2..2147483647)
1477 INTEGER: The number of impressions sent to the device for
1478 the job so far.
1479
1480 impressionsInterpreted(112), Integer32 (-2..2147483647)
1481 INTEGER: The number of impressions interpreted for the job
1482 so far.
1483
1484 impressionsCompletedCurrentCopy(113),
1485 Integer32 (-2..2147483647)
1486 INTEGER: The number of impressions completed by the device
1487 for the current copy of the current document so far. For
1488 printing, the impressions completed includes interpreting,
1489 marking, and stacking the output. For other types of job
1490 services, the number of impressions completed includes the
1491 number of impressions processed.
1492
1493 This value SHALL be reset to 0 for each document in the job
1494 and for each document copy.
1495
1496 fullColorImpressionsCompleted(114), Integer32 (-2..2147483647)
1497 INTEGER: The number of full color impressions completed by
1498 the device for this job so far. For printing, the
1499 impressions completed includes interpreting, marking, and
1500 stacking the output. For other types of job services, the
1501 number of impressions completed includes the number of
1502 impressions processed. Full color impressions are typically
1503 defined as those requiring 3 or more colorants, but this
1504 MAY vary by implementation. In any case, the value of this
1505 attribute counts by 1 for each side that has full color,
1506 not by the number of colors per side (and the other
1507 impression counters are incremented, except
1508 highlightColorImpressionsCompleted(115)).
1509

1510 highlightColorImpressionsCompleted(115),
1511 Integer32 (-2..2147483647)
1512 INTEGER: The number of highlight color impressions
1513 completed by the device for this job so far. For printing,
1514 the impressions completed includes interpreting, marking,
1515 and stacking the output. For other types of job services,
1516 the number of impressions completed includes the number of
1517 impressions processed. Highlight color impressions are
1518 typically defined as those requiring black plus one other
1519 colorant, but this MAY vary by implementation. In any
1520 case, the value of this attribute counts by 1 for each side
1521 that has highlight color (and the other impression counters
1522 are incremented, except
1523 fullColorImpressionsCompleted(114)).
1524
1525 ++++++
1526 + Page attributes
1527 +
1528 + See the definition of 'impression', 'sheet', and 'page'
1529 + in Section 2.
1530 ++++++

1531
1532 pagesRequested(130), Integer32 (-2..2147483647)
1533 INTEGER: The number of logical pages requested by the job
1534 to be processed.
1535

1536 pagesCompleted(131), Integer32 (-2..2147483647)
1537 INTEGER: The number of logical pages completed for this
1538 job so far.
1539
1540 For implementations where multiple copies are produced by
1541 the interpreter with only a single pass over the data, the
1542 final value SHALL be equal to the value of the
1543 pagesRequested object. For implementations where multiple
1544 copies are produced by the interpreter by processing the
1545 data for each copy, the final value SHALL be a multiple of
1546 the value of the pagesRequested object.
1547

1548 NOTE - See the impressionsCompletedCurrentCopy and
1549 pagesCompletedCurrentCopy attributes for attributes that
1550 are reset on each document copy.
1551

1552 NOTE - The pagesCompleted object can be used with the
1553 pagesRequested object to provide an indication of the
1554 relative progress of the job, provided that the
1555 multiplicative factor is taken into account for some
1556 implementations of multiple copies.
1557

1558 pagesCompletedCurrentCopy(132), Integer32 (-2..2147483647)
1559 INTEGER: The number of logical pages completed for the
1560 current copy of the document so far. This value SHALL be
1561 reset to 0 for each document in the job and for each
1562 document copy.
1563
1564 ++++++
1565 + Sheet attributes
1566 +
1567 + See the definition of 'impression', 'sheet', and 'page'
1568 + in Section 2.
1569 ++++++

1570
1571 sheetsRequested(150), Integer32 (-2..2147483647)
1572 INTEGER: The total number of medium sheets requested to be
1573 produced for this job.
1574
1575 Unlike the jmJobKOctetsPerCopyRequested and
1576 jmJobImpressionsPerCopyRequested attributes, the
1577 sheetsRequested(150) attribute SHALL include the
1578 multiplicative factor contributed by the number of copies
1579 and so is the total number of sheets to be produced by the
1580 job, as opposed to the size of the document(s) submitted.
1581

1582 sheetsCompleted(151), Integer32 (-2..2147483647)
1583 INTEGER: The total number of medium sheets that have
1584 completed marking and stacking for the entire job so far
1585 whether those sheets have been processed on one side or on
1586 both.
1587

1588 sheetsCompletedCurrentCopy(152), Integer32 (-2..2147483647)
1589 INTEGER: The number of medium sheets that have completed
1590 marking and stacking for the current copy of a document in
1591 the job so far whether those sheets have been processed on
1592 one side or on both.
1593
1594 The value of this attribute SHALL be 0 before the job
1595 starts processing and SHALL be reset to 1 after the first
1596 sheet of each document and document copy in the job is
1597 processed and stacked.
1598


```

1599 ++++++
1600 + Resources attributes (requested and consumed)
1601 +
1602 + Pairs of these attributes can be used by monitoring
1603 + applications to show an indication of relative usage to
1604 + users, i.e., a 'thermometer'.
1605 ++++++
1606
1607 mediumRequested(170),                               JmMediumTypeTC
1608                                                         AND/OR
1609                                                         JmJobStringTC (SIZE(0..63))
1610     INTEGER: MULTI-ROW: The type
1611     AND/OR
1612     OCTETS: MULTI-ROW: the name of the medium that is
1613     required by the job.
1614
1615     NOTE - The name (JmJobStringTC) values correspond to the
1616     name values of the prtInputMediaName object in the Printer
1617     MIB [print-mib] and the name, size, and input tray values
1618     of the IPP 'media' attribute [ipp-model].
1619
1620 mediumConsumed(171),                               Integer32 (-2..2147483647)
1621                                                         AND
1622                                                         JmJobStringTC (SIZE(0..63))
1623     INTEGER: MULTI-ROW: The number of sheets
1624     AND
1625     OCTETS: MULTI-ROW: the name of the medium that has been
1626     consumed so far whether those sheets have been processed on
1627     one side or on both.
1628
1629     This attribute SHALL have both Integer32 and OCTET STRING
1630     (represented as JmJobStringTC) values.
1631
1632     NOTE - The name (JmJobStringTC) values correspond to the
1633     name values of the prtInputMediaName object in the Printer
1634     MIB [print-mib] and the name, size, and input tray values
1635     of the IPP 'media' attribute [ipp-model].
1636
1637 colorantRequested(172),                             Integer32 (-2..2147483647)
1638                                                         AND/OR
1639                                                         JmJobStringTC (SIZE(0..63))
1640     INTEGER: MULTI-ROW: The index (prtMarkerColorantIndex) in
1641     the Printer MIB[print-mib]
1642     AND/OR
1643     OCTETS: MULTI-ROW: the name of the colorant requested.
1644
1645     NOTE - The name (JmJobStringTC) values correspond to the
1646     name values of the prtMarkerColorantValue object in the
1647     Printer MIB. Examples are: red, blue.
1648
1649 colorantConsumed(173),                             Integer32 (-2..2147483647)
1650                                                         AND/OR

```


1651 JmJobStringTC (SIZE(0..63))
 1652 INTEGER: MULTI-ROW: The index (prtMarkerColorantIndex) in
 1653 the Printer MIB[print-mib]
 1654 AND/OR
 1655 OCTETS: MULTI-ROW: the name of the colorant consumed.
 1656
 1657 NOTE - The name (JmJobStringTC) values correspond to the
 1658 name values of the prtMarkerColorantValue object in the
 1659 Printer MIB. Examples are: red, blue
 1660
 1661 mediumTypeConsumed(174), Integer32 (-2..2147483647)
 1662 AND
 1663 JmJobStringTC (SIZE(0..63))
 1664 INTEGER: MULTI-ROW: The number of sheets of the indicated
 1665 medium type that has been consumed so far whether those
 1666 sheets have been processed on one side or on both
 1667 AND
 1668 OCTETS: MULTI-ROW: the name of that medium type.
 1669
 1670 This attribute SHALL have both Integer32 and OCTET STRING
 1671 (represented as JmJobStringTC) values.
 1672
 1673 NOTE - The type name (JmJobStringTC) values correspond to
 1674 the type name values of the prtInputMediaType object in the
 1675 Printer MIB [print-mib]. Values are: 'stationery',
 1676 'transparency', 'envelope', etc. These medium type names
 1677 correspond to the enum values of JmMediumTypeTC used in the
 1678 mediumRequested attribute.
 1679
 1680 mediumSizeConsumed(175), Integer32 (-2..2147483647)
 1681 AND
 1682 JmJobStringTC (SIZE(0..63))
 1683 INTEGER: MULTI-ROW: The number of sheets of the indicated
 1684 medium size that has been consumed so far whether those
 1685 sheets have been processed on one side or on both
 1686 AND
 1687 OCTETS: MULTI-ROW: the name of that medium size.
 1688
 1689 This attribute SHALL have both Integer32 and OCTET STRING
 1690 (represented as JmJobStringTC) values.
 1691
 1692 NOTE - The size name (JmJobStringTC) values correspond to
 1693 the size name values in the Printer MIB [print-mib]
 1694 Appendix B. These size name values are also a subset of
 1695 the keyword values defined by [ipp-model] for the 'media'
 1696 Job Template attribute. Values are: 'letter', 'a', 'iso-
 1697 a4', 'jis-b4', etc.
 1698
 1699
 1700 ++++++
 1701 + Time attributes (set by server or device)
 1702 +

1703 + This section of attributes are ones that are set by the
1704 + server or device that accepts jobs. Two forms of time are
1705 + provided. Each form is represented in a separate attribute.
1706 + See section 3.1.2 and section 3.1.3 for the
1707 + conformance requirements for time attribute for agents and
1708 + monitoring applications, respectively. The two forms are:
1709 +
1710 + 'DateAndTime' is an 8 or 11 octet binary encoded year,
1711 + month, day, hour, minute, second, deci-second with
1712 + optional offset from UTC. See SNMPv2-TC [SMIV2-TC].
1713 +
1714 + NOTE: 'DateAndTime' is not printable characters; it is
1715 + binary.
1716 +
1717 + 'JmTimeStampTC' is the time of day measured in the number of
1718 + seconds since the system was booted.
1719 ++++++
1720
1721 jobSubmissionToServerTime(190), JmTimeStampTC
1722 AND/OR
1723 DateAndTime
1724 INTEGER: Configuration 3 only: The time
1725 AND/OR
1726 OCTETS: the date and time that the job was submitted to
1727 the server (as distinguished from the device which uses
1728 jobSubmissionTime).
1729
1730 jobSubmissionTime(191), JmTimeStampTC
1731 AND/OR
1732 DateAndTime
1733 INTEGER: Configurations 1, 2, and 3: The time
1734 AND/OR
1735 OCTETS: the date and time that the job was submitted to
1736 the server or device to which the agent is providing
1737 access.
1738

1739 jobStartedBeingHeldTime(192), JmTimeStampTC
 1740 AND/OR
 1741 DateAndTime
 1742 INTEGER: The time
 1743 AND/OR
 1744 OCTETS: the date and time that the job last entered the
 1745 pendingHeld state. If the job has never entered the
 1746 pendingHeld state, then the value SHALL be '0' or the
 1747 attribute SHALL not be present in the table.
 1748
 1749 jobStartedProcessingTime(193), JmTimeStampTC
 1750 AND/OR
 1751 DateAndTime
 1752 INTEGER: The time
 1753 AND/OR
 1754 OCTETS: the date and time that the job started processing.
 1755
 1756 jobCompletionTime(194), JmTimeStampTC
 1757 AND/OR
 1758 DateAndTime
 1759 INTEGER: The time
 1760 AND/OR
 1761 OCTETS: the date and time that the job entered the
 1762 completed, canceled, or aborted state.
 1763
 1764 jobProcessingCPUtime(195) Integer32 (-2..2147483647)
 1765 UNITS 'seconds'
 1766 INTEGER: The amount of CPU time in seconds that the job
 1767 has been in the processing state. If the job enters the
 1768 processingStopped state, that elapsed time SHALL not be
 1769 included. In other words, the jobProcessingCPUtime value
 1770 SHOULD be relatively repeatable when the same job is
 1771 processed again on the same device.

1772

1773 3.4 Monitoring Job Progress

1774 There are a number of objects and attributes for monitoring the
 1775 progress of a job. These objects and attributes count the number of K
 1776 octets, impressions, sheets, and pages requested or completed. For
 1777 impressions and sheets, "completed" ~~SHALL~~ means stacked, unless the
 1778 implementation is unable to detect when each sheet is stacked, in which
 1779 case stacked is approximated when processing of each sheet completes.
 1780 There are objects and attributes for the overall job and for the
 1781 current copy of the document currently being stacked. For the latter,
 1782 the rate at which the various objects and attributes count depends on
 1783 the sheet and document collation of the job.

1784 Job Collation included sheet collation and document collation. Sheet
 1785 collation is defined to be the ordering of sheets within a document
 1786 copy. Document collation is defined to be ordering of document copies

1787 within a multi-document job. There are three types of job collation
1788 (see terminology definitions in Section 2):

1789 1. uncollatedSheets(3) - No collation of the sheets within each
1790 document copy, i.e., each sheet of a document that is to
1791 produce multiple copies is replicated before the next sheet in
1792 the document is processed and stacked. If the device has an
1793 output bin collator, the uncollatedSheets(3) value may actually
1794 produce collated sheets as far as the user is concerned (in the
1795 output bins). However, when the job collation is the
1796 'uncollatedSheets(3)' value, job progress is indistinguishable
1797 to a monitoring application between a device that has an output
1798 bin collator and one that does not.

1799 2. collatedDocuments(4) - Collation of the sheets within each
1800 document copy is performed within the printing device by making
1801 multiple passes over either the source or an intermediate
1802 representation of the document. In addition, when there are
1803 multiple documents per job, the i'th copy of each document is
1804 stacked before the j'th copy of each document, i.e., the
1805 documents are collated within each job copy. For example, if a
1806 job is submitted with documents, A and B, the job is made
1807 available to the end user as: A, B, A, B, The
1808 'collatedDocuments(4)' value corresponds to the IPP [ipp-model]
1809 'separate-documents-collated-copies' value of the "multiple-
1810 document-handling" attribute.

1811
1812 If jobCopiesRequested or documentCopiesRequested = 1, then
1813 jobCollationType is defined as 4.

1814 3. uncollatedDocuments(5) - Collation of the sheets within each
1815 document copy is performed within the printing device by making
1816 multiple passes over either the source or an intermediate
1817 representation of the document. In addition, when there are
1818 multiple documents per job, all copies of the first document in
1819 the job are stacked before the any copied of the next document
1820 in the job, i.e., the documents are uncollated within the job.
1821 For example, if a job is submitted with documents, A and B, the
1822 job is mad available to the end user as: A, A, ..., B, B,
1823 The 'uncollatedDocuments(5)' value corresponds to the IPP [ipp-
1824 model] 'separate-documents-uncollated-copies' value of the
1825 "multiple-document-handling" attribute.

1826 Consider the following four variables that are used to monitor the
1827 progress of a job's impressions:

1828 1. jmJobImpressionsCompleted - counts the total number of
1829 impressions stacked for the job

1830 2. impressionsCompletedCurrentCopy - counts the number of
1831 impressions stacked for the current document copy

1832 3. sheetCompletedCopyNumber - identifies the number of the copy
1833 for the current document being stacked where the first copy is
1834 1.

1835 4. sheetCompletedDocumentNumber - identifies the current document
1836 within the job that is being stacked where the first document
1837 in a job is 1. NOTE: this attribute SHOULD NOT be implemented
1838 for implementations that only support one document per job.

1839 For each of the three types of job collation, a job with three copies
1840 of two documents (1, 2), where each document consists of 3 impressions,
1841 the four variables have the following values as each sheet is stacked
1842 for one-sided printing:

1843 Job Collation Type = uncollatedSheets(3)

1844

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

1845

1846 Job Collation Type = collatedDocuments(4)

1847

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

1848

1849 Job Collation Type = uncollatedDocuments(5)
1850

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

1851

1852 **3.5 Job Identification**

1853 There are a number of attributes that permit a user, operator or system
1854 administrator to identify jobs of interest, such as jobURI, jobName,
1855 jobOriginatingHost, etc. In addition, there is a jmJobSubmissionID
1856 object that is a text string table index. Being a table index allows a
1857 monitoring application to quickly locate and identify a particular job
1858 of interest that was submitted from a particular client by the user
1859 invoking the monitoring application without having to scan the entire
1860 job table. The Job Monitoring MIB needs to provide for identification
1861 of the job at both sides of the job submission process. The primary
1862 identification point is the client side. The jmJobSubmissionID allows
1863 the monitoring application to identify the job of interest from all the
1864 jobs currently "known" by the server or device. The value of
1865 jmJobSubmissionID can be assigned by either the client's local system
1866 or a downstream server or device. The point of assignment depends on
1867 the job submission protocol in use.

1868 The server/device-side identifier, called the jmJobIndex object, SHALL
1869 be assigned by the SNMP Job Monitoring MIB agent when the server or
1870 device accepts the jobs from submitting clients. The jmJobIndex object
1871 allows the interested party to obtain all objects desired that relate
1872 to a particular job. See Section 3.2, entitled 'The Job Tables and the

1873 Oldest Active and Newest Active Indexes' for the specification of how
1874 the agent SHALL assign the jmJobIndex values.

1875 The MIB provides a mapping table that maps each jmJobSubmissionID value
1876 to a corresponding jmJobIndex value generated by the agent, so that an
1877 application can determine the correct value for the jmJobIndex value
1878 for the job of interest in a single Get operation, given the Job
1879 Submission ID. See the jmJobIDGroup.

1880 In some configurations there may be more than one application program
1881 that monitors the same job when the job passes from one network entity
1882 to another when it is submitted. See configuration 3. When there are
1883 multiple job submission IDs, each entity MAY supply an appropriate
1884 jmJobSubmissionID value. In this case there would be a separate entry
1885 in the jmJobSubmissionID table, one for each jmJobSubmissionID. All
1886 entries would map to the same jmJobIndex that contains the job data.
1887 When the job is deleted, it is up to the agent to remove all entries
1888 that point to the job from the jmJobSubmissionID table as well.

1889 The jobName attribute provides a name that the user supplies as a job
1890 attribute with the job. The jobName attribute is not necessarily
1891 unique, even for one user, let alone across users.

1892 3.6 Internationalization Considerations

1893 This section describes the internationalization considerations included
1894 in this MIB.

1895 3.6.1 Text generated by the server or device

1896 There are a few objects and attributes generated by the server or
1897 device that SHALL be represented using the Universal Multiple-Octet
1898 Coded Character Set (UCS) [ISO-10646]. These objects and attributes
1899 are always supplied (if implemented) by the agent, not by the job
1900 submitting client:

- 1901 1. jmGeneralJobSetName object
- 1902 2. processingMessage(6) attribute
- 1903 3. physicalDevice(32) (name value) attribute

1904 The character encoding scheme for representing these objects and
1905 attributes SHALL be UTF-8 as recommended by RFC 2130 [RFC 2130] and the
1906 "IETF Policy on Character Sets and Language" [char-set policy]. The
1907 'JmUTF8StringTC' textual convention is used to indicate UTF-8 text
1908 strings.

1909 NOTE - For strings in 7-bit US-ASCII, there is no impact since the UTF-
1910 8 representation of 7-bit ASCII is identical to the US-ASCII [US-ASCII]
1911 encoding.

1912 The text contained in the processingMessage(6) attribute is generated
1913 by the server/device. The natural language for the

1914 processingMessage(6) attribute is identified by the
1915 processingMessageNaturalLangTag(7) attribute. The
1916 processingMessageNaturalLangTag(7) attribute uses the
1917 JmNaturalLanguageTagTC textual convention which SHALL conform to the
1918 language tag mechanism specified in RFC 1766 [RFC-1766]. The
1919 JmNaturalLanguageTagTC value is the same as the IPP [IPP-model]
1920 'naturalLanguage' attribute syntax. RFC 1766 specifies that a US-ASCII
1921 string consisting of the natural language followed by an optional
1922 country field. Both fields use the same two-character codes from ISO
1923 639 [ISO-639] and ISO 3166 [ISO-3166], respectively, that are used in
1924 the Printer MIB for identifying language and country.

1925 Examples of the values of the processingMessageNaturalLangTag(7)
1926 attribute include:

- 1927 1. 'en' for English
- 1928 2. 'en-us' for US English
- 1929 3. 'fr' for French
- 1930 4. 'de' for German

1931 3.6.2 Text supplied by the job submitter

1932 All of the objects and attributes represented by the 'JmJobStringTC'
1933 textual-convention are either (1) supplied in the job submission
1934 protocol by the client that submits the job to the server or device or
1935 (2) are defaulted by the server or device if the job submitting client
1936 does not supply values. The agent SHALL represent these objects and
1937 attributes in the MIB either (1) in the coded character set as they
1938 were submitted or (2) MAY convert the coded character set to another
1939 coded character set or encoding scheme. In any case, the resulting
1940 coded character set representation SHOULD be UTF-8 [UTF-8], but SHALL
1941 be one in which the code positions from 0 to 31 ~~SHALL-is not be-used~~,
1942 32 to 127 ~~SHALL-beis~~ US-ASCII [US-ASCII], 127 ~~SHALL-beis not~~ unused,
1943 and the remaining code positions 128 to 255 ~~SHALL-represent~~ single-byte
1944 or multi-byte graphic characters structured according to ISO 2022 [ISO
1945 2022] or ~~SHALL-beare~~ unused.

1946 The coded character set SHALL be one of the ones registered with IANA
1947 [IANA] and SHALL be identified by the jobCodedCharSet attribute in the
1948 jmJobAttributeTable for the job. If the agent does not know what coded
1949 character set was used by the job submitting client, the agent SHALL
1950 either (1) return the 'unknown(2)' value for the jobCodedCharSet
1951 attribute or (2) not return the jobCodedCharSet attribute for the job.

1952 Examples of coded character sets which meet this criteria for use as
1953 the value of the jobCodedCharSet job attribute are: US-ASCII [US-
1954 ASCII], ISO 8859-1 (Latin-1) [ISO 8859-1], any ISO 8859-n, HP Roman8,
1955 IBM Code Page 850, Windows Default 8-bit set, UTF-8 [UTF-8], US-ASCII
1956 plus JIS X0208-1990 Japanese [JIS X0208], US-ASCII plus GB2312-1980 PRC
1957 Chinese [GB2312]. See the IANA registry of coded character sets [IANA
1958 charsets].

1959 Examples of coded character sets which do not meet this criteria are:
1960 national 7-bit sets conforming to ISO 646 (except US-ASCII), EBCDIC,
1961 and ISO 10646 (Unicode) [ISO-10646]. In order to represent Unicode
1962 characters, the UTF-8 [UTF-8] encoding scheme SHALL be used which has
1963 been assigned the MIBenum value of '106' by IANA.

1964 The jobCodedCharSet attribute uses the imported 'CodedCharSet' textual-
1965 convention from the Printer MIB [printmib].

1966 The natural language for attributes represented by the textual-
1967 convention JmJobStringTC ~~SHALL-beis~~ identified either (1) by the
1968 jobNaturalLanguageTag(9) attribute or ~~SHALL-beis~~ keywords in US-English
1969 (as in IPP). A monitoring application SHOULD attempt to localize
1970 keywords into the language of the user by means of some lookup
1971 mechanism. If the keyword value is not known to the monitoring
1972 application, the monitoring application SHOULD assume that the value is
1973 in the natural language specified by the job's jobNaturalLanguageTag(9)
1974 attribute and SHOULD present the value to its user as is. The
1975 jobNaturalLanguageTag(9) attribute value SHALL have the same syntax and
1976 semantics as the processingMessageNaturalLangTag(7) attribute, except
1977 that the jobNaturalLanguageTag(9) attribute identifies the natural
1978 language of attributes supplied by the job submitter instead of the
1979 natural language of the processingMessage(6) attribute. See Section
1980 3.6.1.

1981 3.6.3 'DateAndTime' for representing the date and time

1982 This MIB also contains objects that are represented using the
1983 DateAndTime textual convention from SMIV2 [SMIV2-TC]. The job
1984 management application SHALL display such objects in the locale of the
1985 user running the monitoring application.

1986 3.7 IANA and PWG Registration Considerations

1987 This MIB does not require any additional registration schemes for IANA,
1988 but does depend on registration schemes that other Internet standards
1989 track specifications have set up. The names of these IANA registration
1990 assignments under the /in-notes/iana/assignments/ path:

1991 1. printer-language-numbers - used as enums in the documentFormat(38)
1992 attribute

1993 2. media-types - uses as keywords in the documentFormat(38) attribute

1994 3. character-sets - used as enums in the jobCodedCharSet(8) attribute

1995 The Printer Working Group (PWG) will handle registration of additional
1996 enums after approving this standard, according to the procedures
1997 described in this section:

1998

1999 3.7.1 PWG Registration of enums

2000 This specification uses textual conventions to define enumerated values
2001 (enums) and bit values. Enumerations (enums) and bit values are sets
2002 of symbolic values defined for use with one or more objects or
2003 attributes. All enumeration sets and bit value sets are assigned a
2004 symbolic data type name (textual convention). As a convention the
2005 symbolic name ends in "TC" for textual convention. These enumerations
2006 are defined at the beginning of the MIB module specification.

2007 The PWG has defined several type of enumerations for use in the Job
2008 Monitoring MIB and the Printer MIB[print-mib]. These types differ in
2009 the method employed to control the addition of new enumerations.
2010 Throughout this document, references to "type n enum", where n can be
2011 1, 2 or 3 can be found in the various tables. The definitions of these
2012 types of enumerations are:

2013 3.7.1.1 Type 1 enumerations

2014 Type 1 enumeration: All the values are defined in the Job Monitoring
2015 MIB specification (RFC for the Job Monitoring MIB). Additional
2016 enumerated values require a new RFC.

2017 There are no type 1 enums in the current draft.

2018 3.7.1.2 Type 2 enumerations

2019 Type 2 enumeration: An initial set of values are defined in the Job
2020 Monitoring MIB specification. Additional enumerated values are
2021 registered with the PWG.

2022 The following type 2 enums are contained in the current draft :

- 2023 1. JmUTF8StringTC
- 2024 2. JmJobStringTC
- 2025 3. JmNaturalLanguageTagTC
- 2026 4. JmTimeStampTC
- 2027 5. JmFinishingTC [same enum values as IPP "finishing" attribute]
- 2028 6. JmPrintQualityTC [same enum values as IPP "print-quality"
2029 attribute]
- 2030 7. JmTonerEconomyTC
- 2031 8. JmMediumTypeTC
- 2032 9. JmJobSubmissionIDTypeTC
- 2033 10. JmJobCollationTypeTC
- 2034 11. JmJobStateTC [same enum values as IPP "job-state" attribute]
- 2035 12. JmAttributeTypeTC

2036 For those textual conventions that have the same enum values as the
2037 indicated IPP Job attribute ~~SHALL-beare~~ simultaneously registered by
2038 the PWG for use with IPP [ipp-model] and the Job Monitoring MIB.

2039 3.7.1.3 Type 3 enumeration

2040 Type 3 enumeration: An initial set of values are defined in the Job
2041 Monitoring MIB specification. Additional enumerated values are
2042 registered through the PWG without PWG review.

2043 There are no type 3 enums in the current draft.

2044 3.7.2 PWG Registration of type 2 bit values

2045 This draft contains the following type 2 bit value textual-conventions:

- 2046 1. JmJobServiceTypesTC
- 2047 2. JmJobStateReasons1TC
- 2048 3. JmJobStateReasons2TC
- 2049 4. JmJobStateReasons3TC
- 2050 5. JmJobStateReasons4TC

2051 These textual-conventions are defined as bits in an Integer so that
2052 they can be used with SNMPv1 SMI. The jobStateReasonsN (N=1..4)
2053 attributes are defined as bit values using the corresponding
2054 JmJobStateReasonsMTC textual-conventions.

2055 The registration of JmJobServiceTypesTC and JmJobStateReasonsMTC bit
2056 values ~~SHALL~~ follow the procedures for a type 2 enum as specified in
2057 Section 3.7.1.2.

2058 3.7.3 PWG Registration of Job Submission Id Formats

2059 In addition to enums and bit values, this specification assigns a
2060 single ASCII digit or letter to various job submission ID formats. See
2061 the JmJobSubmissionIDTypeTC textual-convention and the object. The
2062 registration of JobSubmissionID format numbers ~~SHALL~~ follows the
2063 procedures for a type 2 enum as specified in Section 3.7.1.2.

2064 3.7.4 PWG Registration of MIME types/sub-types for document-formats

2065 The documentFormat(38) attribute has MIME type/sub-type values for
2066 indicating document formats which IANA registers as "media type" names.
2067 The values of the documentFormat(38) attribute are the same as the
2068 corresponding Internet Printing Protocol (IPP) "document-format" Job
2069 attribute values [ipp-model].

2070 **3.8 Security Considerations**

2071 3.8.1 Read-Write objects

2072 All objects are read-only, greatly simplifying the security
2073 considerations. If another MIB augments this MIB, that MIB might
2074 accept SNMP Write operations to objects in that MIB whose effect is to
2075 modify the values of read-only objects in this MIB. However, that MIB
2076 SHALL have to support the required access control in order to achieve
2077 security, not this MIB.

2078 3.8.2 Read-Only Objects In Other User's Jobs

2079 The security policy of some sites MAY be that unprivileged users can
2080 only get the objects from jobs that they submitted, plus a few minimal
2081 objects from other jobs, such as the jmJobKOctetsPerCopyRequested and
2082 jmJobKOctetsProcessed objects, so that a user can tell how busy a
2083 printer is. Other sites MAY allow all unprivileged users to see all
2084 objects of all jobs. This MIB does not require, nor does it specify
2085 how, such restrictions would be implemented. A monitoring application
2086 SHOULD enforce the site security policy with respect to returning
2087 information to an unprivileged end user that is using the monitoring
2088 application to monitor jobs that do not belong to that user, i.e., the
2089 jmJobOwner object in the jmJobTable does not match the user's user
2090 name.

2091 An operator is a privileged user that would be able to see all objects
2092 of all jobs, independent of the policy for unprivileged users.

2093 **3.9 Notifications**

2094 This MIB does not specify any notifications. For simplicity,
2095 management applications are expected to poll for status. The
2096 jmGeneralJobPersistence and jmGeneralAttributePersistence objects
2097 assist an application to determine the polling rate. The resulting
2098 network traffic is not expected to be significant.

2099 4 MIB specification

2100 The following pages constitute the actual Job Monitoring MIB.


```
2101 Job-Monitoring-MIB DEFINITIONS ::= BEGIN
2102
2103 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

2104
2105 -- Use the enterprises arc assigned to the PWG which is pwg(2699).
2106 -- Group all PWG mibs under mibs(1).
2107
2108 jobmonMIB MODULE-IDENTITY
2109     LAST-UPDATED "98100202030000Z"
2110     ORGANIZATION "Printer Working Group (PWG)"
2111     CONTACT-INFO
2112         "Tom Hastings
2113         Postal:  Xerox Corp.
2114                 Mail stop ESAE-231
2115                 701 S. Aviation Blvd.
2116                 El Segundo, CA 90245
2117
2118         Tel:      (301)333-6413
2119         Fax:      (301)333-5514
2120         E-mail:   hastings@cpl0.es.xerox.com
2121
2122         Send questions and comments to the Printer Working Group (PWG)
2123         using the Job Monitoring Project (JMP) Mailing List:
2124         jmp@pwg.org
2125
2126         For further information, including how to subscribe to the
2127         jmp mailing list, access the PWG web page under 'JMP':
2128
2129         http://www.pwg.org/
2130
2131         Implementers of this specification are encouraged to join the
2132         jmp mailing list in order to participate in discussions on any
2133         clarifications needed and registration proposals being reviewed
2134         in order to achieve consensus."
2135     DESCRIPTION
2136         "The MIB module for monitoring job in servers, printers, and
2137         other devices.
2138
2139         Version: 1.20"
2140     ::= { enterprises pwg(2699) mibs(1) jobmonMIB(1) }
```

```
2141
2142 -- Textual conventions for this MIB module
2143
2144 JmUTF8StringTC ::= TEXTUAL-CONVENTION
2145     DISPLAY-HINT "255a"
2146     STATUS      current
2147     DESCRIPTION
2148         "To facilitate internationalization, this TC represents
2149         information taken from the ISO/IEC IS 10646-1 character set,
2150         encoded as an octet string using the UTF-8 character encoding
2151         scheme."
2152     REFERENCE
2153         
2154
2155         See section 3.6.1, entitled: 'Text generated by the server or
2156         device'."
2157     SYNTAX      OCTET STRING (SIZE (0..63))
2158
2159
2160
2161
2162 JmJobStringTC ::= TEXTUAL-CONVENTION
2163     STATUS      current
2164     DESCRIPTION
2165         "To facilitate internationalization, this TC represents
2166         information using any coded character set registered by IANA as
2167         specified in section 3.7. While it is recommended that the
2168         coded character set be UTF-8 [UTF-8], the actual coded
2169         character set SHALL be indicated by the value of the
2170         jobCodedCharSet(8) attribute for the job."
2171     REFERENCE
2172         
2173
2174         See section 3.6.2, entitled: 'Text supplied by the job
2175         submitter'."
2176     SYNTAX      OCTET STRING (SIZE (0..63))
2177
2178
2179
2180
2181 JmNaturalLanguageTagTC ::= TEXTUAL-CONVENTION
2182     STATUS      current
2183     DESCRIPTION
2184         "An IETF RFC 1766-compliant 'language tag', with zero or more
2185         sub-tags that identify a natural language. While RFC 1766
2186         specifies that the US-ASCII values are case-insensitive, this
2187         MIB specification requires that all characters SHALL be lower
2188         case in order to simplify comparing by management
2189         applications."
2190     REFERENCE
2191         
2192
```

2193 See section 3.6.1, entitled: 'Text generated by the server or
2194 device' and section 3.6.2, entitled: 'Text supplied by the job
2195 submitter'."

2196 SYNTAX OCTET STRING (SIZE (0..63))
2197
2198

2199 JmTimeStampTC ::= TEXTUAL-CONVENTION
2200 STATUS current
2201 DESCRIPTION
2202 "The simple time at which an event took place. The units ~~SHALL~~
2203 ~~beare~~ in seconds since the system was booted.
2204

2205 NOTE - JmTimeStampTC is defined in units of seconds, rather
2206 than 100ths of seconds, so as to be simpler for agents to
2207 implement (even if they have to implement the 100ths of a
2208 second to comply with implementing sysUpTime in MIB-II[mib-
2209 II].)
2210

2211 NOTE - JmTimeStampTC is defined as an Integer32 so that it can
2212 be used as a value of an attribute, i.e., as a value of the
2213 jmAttributeValueAsInteger object. The TimeStamp textual-
2214 convention defined in SNMPv2-TC [SMIV2-TC] is defined as an
2215 APPLICATION 3 IMPLICIT INTEGER tag, not an Integer32 which is
2216 defined in SNMPv2-SMI [SMIV2-TC] as UNIVERSAL 2 IMPLICIT
2217 INTEGER, so cannot be used in this MIB as one of the values of
2218 jmAttributeValueAsInteger."
2219 SYNTAX INTEGER (0..2147483647)
2220
2221
2222
2223

2224 JmJobSourcePlatformTypeTC ::= TEXTUAL-CONVENTION
2225 STATUS current
2226 DESCRIPTION
2227 "The source platform type that can submit jobs to servers or
2228 devices in any of the 3 configurations."
2229 ~~REFERENCE~~
2230 ~~"~~
2231

2232 This is a type 2 enumeration. See Section 3.7.1.2. See also
2233 IANA operating-system-names registry."
2234 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

```
2235     }
2236     sptWindows(11),      -- Windows
    sptNetWare(12)        -- NetWare
```

```
2237
2238 JmFinishingTC ::= TEXTUAL-CONVENTION
2239     STATUS      current
2240     DESCRIPTION
2241         "The type of finishing operation.
2242
2243         These values are the same as the enum values of the IPP
2244         'finishings' attribute.  See Section 3.7.1.2.
2245
2246         other(1),
2247             Some other finishing operation besides one of the specified
2248             or registered values.
2249
2250         unknown(2),
2251             The finishing is unknown.
2252
2253         none(3),
2254             Perform no finishing.
2255
2256         staple(4),
2257             Bind the document(s) with one or more staples. The exact
2258             number and placement of the staples is site-defined.
2259
2260         punch(5),
2261             This value indicates that holes are required in the
2262             finished document. The exact number and placement of the
2263             holes is site-defined. The punch specification MAY be
2264             satisfied (in a site- and implementation-specific manner)
2265             either by drilling/punching, or by substituting pre-drilled
2266             media.
2267
2268         cover(6),
2269             This value is specified when it is desired to select a non-
2270             printed (or pre-printed) cover for the document. This does
2271             not supplant the specification of a printed cover (on cover
2272             stock medium) by the document itself.
2273
2274         bind(7)
2275             This value indicates that a binding is to be applied to the
2276             document; the type and placement of the binding is product-
2277             specific."  
2278     REFERENCE  
2279     "  
2280  
2281     This is a type 2 enumeration.  See Section 3.7.1.2."  
2282     SYNTAX      INTEGER {  
2283         other(1),  
2284         unknown(2),  
2285         none(3),  
2286         staple(4),  
2287         punch(5),  
2288         cover(6),
```

```
2289     bind(7)
2290   }
2291
2292
2293 JmPrintQualityTC ::= TEXTUAL-CONVENTION
2294     STATUS      current
2295     DESCRIPTION
2296       "Print quality settings.
2297
2298       These values are the same as the enum values of the IPP 'print-
2299       quality' attribute. See Section 3.7.1.2."u
2300 REFERENCE
2301 u
2302
2303       This is a type 2 enumeration. See Section 3.7.1.2."
2304 SYNTAX      INTEGER {
2305     other(1),      -- Not one of the specified or registered
2306                   -- values.
2307     unknown(2),   -- The actual value is unknown.
2308     draft(3),     -- Lowest quality available on the printer.
2309     normal(4),    -- Normal or intermediate quality on the
2310                   -- printer.
2311     high(5)       -- Highest quality available on the printer.
2312   }
2313
2314
2315 JmPrinterResolutionTC ::= TEXTUAL-CONVENTION
2316     STATUS      current
2317     DESCRIPTION
2318       "Printer resolutions.
2319
2320       Nine octets consisting of two 4-octet SIGNED-INTEGERS followed
2321       by a SIGNED-BYTE. The values are the same as those specified
2322       in the Printer MIB [printmib]. The first SIGNED-INTEGER
2323       contains the value of prtMarkerAddressabilityXFeedDir. The
2324       second SIGNED-INTEGER contains the value of
2325       prtMarkerAddressabilityFeedDir. The SIGNED-BYTE contains the
2326       value of prtMarkerAddressabilityUnit.
2327
2328       Note: the latter value is either 3 (tenThousandsOfInches) or 4
2329       (micrometers) and the addressability is in 10,000 units of
2330       measure. Thus the SIGNED-INTEGERS represent integral values in
2331       either dots-per-inch or dots-per-centimeter.
2332
2333       The syntax is the same as the IPP 'printer-resolution'
2334       attribute. See Section 3.7.1.2."
2335 SYNTAX      OCTET STRING (SIZE(9))
2336
2337
```

```
2332
2333 JmTonerEconomyTC ::= TEXTUAL-CONVENTION
2334     STATUS      current
2335     DESCRIPTION
2336         "Toner economy settings."
2337     REFERENCE
2338         
2339
2340         This is a type 2 enumeration.  See Section 3.7.1.2."
2341     SYNTAX      INTEGER {
2342         unknown(2),      -- unknown.
2343         off(3),          -- Off. Normal. Use full toner.
2344         on(4)            -- On. Use less toner than normal.
2345     }
2346 JmBooleanTC ::= TEXTUAL-CONVENTION
2347     STATUS      current
2348     DESCRIPTION
2349         "Boolean true or false value."
2350     REFERENCE
2351         
2352
2353         This is a type 2 enumeration.  See Section 3.7.1.2."
2354     SYNTAX      INTEGER {
2355         unknown(2),      -- unknown.
2356         false(3),        -- FALSE.
2357         true(4)          -- TRUE.
2358     }
2359 JmMediumTypeTC ::= TEXTUAL-CONVENTION
2360     STATUS      current
2361     DESCRIPTION
2362         "Identifies the type of medium.
2363
2364         other(1),
2365             The type is neither one of the values listed in this
2366             specification nor a registered value.
2367
2368         unknown(2),
2369             The type is not known.
2370
2371         stationery(3),
2372             Separately cut sheets of an opaque material.
2373
2374         transparency(4),
2375             Separately cut sheets of a transparent material.
2376
```


2377 envelope(5),
2378 Envelopes that can be used for conventional mailing
2379 purposes.

2380
2381 envelopePlain(6),
2382 Envelopes that are not preprinted and have no windows.
2383
2384 envelopeWindow(7),
2385 Envelopes that have windows for addressing purposes.
2386
2387 continuousLong(8),
2388 Continuously connected sheets of an opaque material
2389 connected along the long edge.
2390
2391 continuousShort(9),
2392 Continuously connected sheets of an opaque material
2393 connected along the short edge.
2394
2395 tabStock(10),
2396 Media with tabs.
2397
2398 multiPartForm(11),
2399 Form medium composed of multiple layers not pre-attached to
2400 one another; each sheet MAY be drawn separately from an
2401 input source.
2402
2403 labels(12),
2404 Label-stock.
2405
2406 multiLayer(13)
2407 Form medium composed of multiple layers which are pre-
2408 attached to one another, e.g. for use with impact
2409 printers."
2410 **REFERENCE**
2411 **"**
2412
2413 This is a type 2 enumeration. See Section 3.7.1.2. These enum
2414 values correspond to the keyword name strings of the
2415 prtInputMediaType object in the Printer MIB [print-mib]. There
2416 is no printer description attribute in IPP/1.0 that represents
2417 these values."
2418 SYNTAX INTEGER {
2419 other(1),
2420 unknown(2),
2421 stationery(3),
2422 transparency(4),
2423 envelope(5),
2424 envelopePlain(6),
2425 envelopeWindow(7),
2426 continuousLong(8),
2427 continuousShort(9),
2428 tabStock(10),
2429 multiPartForm(11),
2430 labels(12),
2431 multiLayer(13)

```

2432     }
2433
2434
2435 JmJobCollationTypeTC ::= TEXTUAL-CONVENTION
2436     STATUS         current
2437     DESCRIPTION
2438         "This value is the type of job collation. Implementations that
2439         don't support multiple documents or don't support multiple
2440         copies SHALL NOT support the uncollatedDocuments(5) value."
2441     REFERENCE
2442         "
2443
2444         This is a type 2 enumeration. See Section 3.7.1.2. See also
2445         Section 3.4, entitled 'Monitoring Job Progress'."
2446     SYNTAX         INTEGER {
2447         other(1),
2448         unknown(2),
2449         uncollatedSheets(3),      -- sheets within each document copy
2450                                   -- are not collated: 1 1 ..., 2 2 ...,
2451                                   -- No corresponding value of IPP
2452                                   -- "multiple-document-handling"
2453         collatedDocuments(4),    -- internal collated sheets,
2454                                   -- documents: A, B, A, B, ...
2455                                   -- Corresponds to IPP "multiple-
2456                                   -- document-handling"='separate-
2457                                   -- documents-collated-copies'
2458         uncollatedDocuments(5)  -- internal collated sheets,
2459                                   -- documents: A, A, ..., B, B, ...
2460                                   -- Corresponds to IPP "multiple-
2461                                   -- document-handling"='separate-
2462                                   -- documents-uncollated-copies'
2463     }
2464
2465
2466 JmJobSubmissionIDTypeTC ::= TEXTUAL-CONVENTION
2467     STATUS         current
2468     DESCRIPTION
2469         "Identifies the format type of a job submission ID.
2470
2471         Each job submission ID is a fixed-length, 48-octet printable
2472         US-ASCII [US-ASCII] coded character string containing no
2473         control characters, consisting of the following fields:
2474
2475         octet 1: The format letter identifying the format. The US-
2476         ASCII characters '0-9', 'A-Z', and 'a-z' are assigned in
2477         order giving 62 possible formats.
2478         octets 2-40: A 39-character, US-ASCII trailing SPACE filled
2479         field specified by the format letter, if the data is less
2480         than 39 ASCII characters.
2481         octets 41-48: A sequential or random US-ASCII number to make
2482         the ID quasi-unique.
2483

```

2484 If the client does not supply a job submission ID in the job
 2485 submission protocol, then the agent SHALL assign a job
 2486 submission ID using any of the standard formats that are
 2487 reserved for the agent. Clients SHALL not use formats that are
 2488 reserved for agents and agents SHALL NOT use formats that are
 2489 reserved for clients, in order to reduce conflicts in ID
 2490 generation. See the description for which formats are reserved
 2491 for clients or for agents.

2492
 2493 Registration of additional formats may be done following the
 2494 procedures described in Section 3.7.3.

2495
 2496 The format values defined at the time of completion of this
 2497 specification are:

2498
 2499 Format

2500 Letter Description

2501 -----

2502 '0' Job Owner generated by the server/device

2503 octets 2-40: The last 39 bytes of the jmJobOwner object.

2504 octets 41-48: The US-ASCII 8-decimal-digit sequential number
 2505 assigned by the agent.

2506 This format is reserved for agents.

2507

2508 NOTE - Clients wishing to use a job submission ID that
 2509 incorporates the job owner, SHALL use format '8', not
 2510 format '0'.

2511

2512 '1' Job Name

2513 octets 2-40: The last 39 bytes of the jobName attribute.

2514 octets 41-48: The US-ASCII 8-decimal-digit random number
 2515 assigned by the client.

2516 This format is reserved for clients.

2517

2518 '2' Client MAC address

2519 octets 2-40: The client MAC address: in hexadecimal with each
 2520 nibble of the 6 octet address being '0'-'9' or 'A' - 'F'
 2521 (uppercase only). Most significant octet first.

2522 octets 41-48: The US-ASCII 8-decimal-digit sequential number
 2523 assigned by the client.

2524 This format is reserved for clients.

2525

2526 '3' Client URL

2527 octets 2-40: The last 39 bytes of the client URL [URI-spec].

2528 octets 41-48: The US-ASCII 8-decimal-digit sequential number
 2529 assigned by the client.

2530 This format is reserved for clients.

2531

2532 '4' Job URI

2533 octets 2-40: The last 39 bytes of the URI [URI-spec] assigned
 2534 by the server or device to the job when the job was
 2535 submitted for processing.

2536 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2537 assigned by the agent.
2538 This format is reserved for agents.
2539

2540 '5' POSIX User Number
2541 octets 2-40: The last 39 bytes of a user number, such as POSIX
2542 user number.
2543 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2544 assigned by the client.
2545 This format is reserved for clients.
2546

2547 '6' User Account Number
2548 octets 2-40: The last 39 bytes of the user account number.
2549 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2550 assigned by the client.
2551 This format is reserved for clients.
2552

2553 '7' DTMF Incoming FAX routing number
2554 octets 2-40: The last 39 bytes of the DTMF incoming FAX
2555 routing number.
2556 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2557 assigned by the client.
2558 This format is reserved for clients.
2559

2560 '8' Job Owner supplied by the client
2561 octets 2-40: The last 39 bytes of the job owner name (that the
2562 agent returns in the jmJobOwner object).
2563 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2564 assigned by the client.
2565 This format is reserved for clients. See format '0' which is
2566 reserved for agents.
2567

2568 '9' Host Name
2569 octets 2-40: The last 39 bytes of the host name with trailing
2570 SPACES that submitted the job to this server/device using a
2571 protocol, such as LPD [RFC-1179] which includes the host
2572 name in the job submission protocol.
2573 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2574 representation of the job id generated by the submitting
2575 server (configuration 3) or the client (configuration 1 and
2576 2), such as in the LPD protocol.
2577 This format is reserved for clients.
2578

2579 'A' AppleTalk Protocol
2580 octets 2-40: Contains the AppleTalk printer name, with the
2581 first character of the name in octet 2. AppleTalk printer
2582 names are a maximum of 31 characters. Any unused portion
2583 of this field shall be filled with spaces.
2584 octets 41-48: '00000XXX', where 'XXX' is the 3-digit US-ASCII
2585 decimal representation of the Connection Id.
2586 This format is reserved for agents.
2587

2588 'B' NetWare PServer
2589 octets 2-40: Contains the Directory Path Name as recorded by
2590 the Novell File Server in the queue directory. If the
2591 string is less than 40 octets, the left-most character in
2592 the string shall appear in octet position 2. Otherwise,
2593 only the last 39 bytes shall be included. Any unused
2594 portion of this field shall be filled with spaces.
2595 octets 41-48: '000XXXXX' The US-ASCII representation of the
2596 Job Number as per the NetWare File Server Queue Management
2597 Services.
2598 This format is reserved for agents.
2599
2600 'C' Server Message Block protocol (SMB)
2601 octets 2-40: Contains a decimal (US-ASCII coded)
2602 representation of the 16 bit SMB Tree Id field, which
2603 uniquely identifies the connection that submitted the job
2604 to the printer. The most significant digit of the numeric
2605 string shall be placed in octet position 2. All unused
2606 portions of this field shall be filled with spaces. The
2607 SMB Tree Id has a maximum value of 65,535.
2608 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2609 representation of the File Handle returned from the device
2610 to the client in response to a Create Print File command.
2611 This format is reserved for agents.
2612
2613 'D' Transport Independent Printer/System Interface (TIP/SI)
2614 octets 2-40: Contains the Job Name from the Job Control-Start
2615 Job (JC-SJ) command. If the Job Name portion is less than
2616 40 octets, the left-most character in the string shall
2617 appear in octet position 2. Any unused portion of this
2618 field shall be filled with spaces. Otherwise, only the
2619 last 39 bytes shall be included.
2620 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2621 representation of the jmJobIndex assigned by the agent.
2622 This format is reserved for agents, since the agent supplies
2623 octets 41-48, though the client supplies the job name. See
2624 format '1' reserved to clients to submit job name ids in
2625 which they supply octets 41-48.
2626
2627 'E' IPDS on the MVS or VSE platform
2628
2629 octets 2-40: Contains bytes 2-27 of the XOH Define Group
2630 Boundary Group ID triplet. Octet position 2 MUST carry the
2631 value x'01'. Bytes 28-40 MUST be filled with spaces.
2632 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2633 representation of the jmJobIndex assigned by the agent.
2634 This format is reserved for agents, since the agent supplies
2635 octets 41-48, though the client supplies the job name.
2636

2637 'F' IPDS on the VM platform
2638 octets 2-40: Contains bytes 2-31 of the XOH Define Group
2639 Boundary Group ID triplet. Octet position 2 MUST carry the
2640 value x'02'. Bytes 32-40 MUST be filled with spaces.
2641 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2642 representation of the jmJobIndex assigned by the agent.
2643 This format is reserved for agents, since the agent supplies
2644 octets 41-48, though the client supplies the file name.
2645
2646 'G' IPDS on the OS/400 platform
2647 octets 2-40: Contains bytes 2-36 of the XOH Define Group
2648 Boundary Group ID triplet. Octet position 2 MUST carry the
2649 value x'03'. Bytes 37-40 MUST be filled with spaces.
2650 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2651 representation of the jmJobIndex assigned by the agent.
2652 This format is reserved for agents, since the agent supplies
2653 octets 41-48, though the client supplies the job name.
2654
2655 NOTE - the job submission id is only intended to be unique
2656 between a limited set of clients for a limited duration of
2657 time, namely, for the life time of the job in the context of
2658 the server or device that is processing the job. Some of the
2659 formats include something that is unique per client and a
2660 random number so that the same job submitted by the same client
2661 will have a different job submission id. For other formats,
2662 where part of the id is guaranteed to be unique for each
2663 client, such as the MAC address or URL, a sequential number
2664 SHOULD suffice for each client (and may be easier for each
2665 client to manage). Therefore, the length of the job submission
2666 id has been selected to reduce the probability of collision to
2667 an extremely low number, but is not intended to be an absolute
2668 guarantee of uniqueness. None-the-less, collisions are
2669 remotely possible, but without bad consequences, since this MIB
2670 is intended to be used only for monitoring jobs, not for
2671 controlling and managing them."
2672 REFERENCE
2673 "
2674
2675 This is like a type 2 enumeration. See section 3.7.3."
2676 SYNTAX OCTET STRING(SIZE(1)) -- ASCII '0'-'9', 'A'-'Z', 'a'-'z'

```

2677
2678 JmJobStateTC ::= TEXTUAL-CONVENTION
2679     STATUS      current
2680     DESCRIPTION
2681         "The current state of the job (pending, processing, completed,
2682         etc.).
2683
2684         The following figure shows the normal job state transitions:
2685
2686                                     +-----> canceled(7)
2687                                     /
2688     +----> pending(3)  -----> processing(5)  -----+-----> completed(9)
2689     |               ^                               ^
2690     |               |                               |
2691     |               v                               v
2692     +----> pendingHeld(4)  processingStopped(6)  ----+
2693     |
2694     |
2695     |
2696     |
2697     |
2698     |
2699     |
2700     |
2701     |
2702     |
2703     |
2704     |
2705     |
2706     |
2707     |
2708     |
2709     |
2710     |
2711     |
2712     |
2713     |
2714     |
2715     |
2716     |
2717     |
2718     |
2719     |
2720     |
2721     |
2722     |
2723     |
2724     |
2725     |
2726     |
2727     |

```

Figure 4 - Normal Job State Transitions

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

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

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

unknown(2),

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

pending(3),

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

pendingHeld(4),

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

2728 JmJobStateReasonsMTC (N=1..4) textual convention for the
2729 specification of each reason.
2730
2731 processing(5),
2732 One or more of:
2733
2734 1. the job is using, or is attempting to use, one or more
2735 purely software processes that are analyzing, creating, or
2736 interpreting a PDL, etc.,
2737
2738 2. the job is using, or is attempting to use, one or more
2739 hardware devices that are interpreting a PDL, making marks
2740 on a medium, and/or performing finishing, such as stapling,
2741 etc.,
2742
2743 OR
2744
2745 3. (configuration 2) the server has made the job ready for
2746 printing, but the output device is not yet printing it,
2747 either because the job hasn't reached the output device or
2748 because the job is queued in the output device or some
2749 other spooler, awaiting the output device to print it.
2750
2751 When the job is in the processing state, the entire job
2752 state includes the detailed status represented in the
2753 device MIB indicated by the hrDeviceIndex value of the
2754 job's physicalDevice attribute, if the agent implements
2755 such a device MIB.
2756
2757 Implementations MAY, though they NEED NOT, include
2758 additional values in the job's jmJobStateReasons1 object to
2759 indicate the progress of the job, such as adding the
2760 jobPrinting value to indicate when the device is actually
2761 making marks on a medium and/or the processingToStopPoint
2762 value to indicate that the server or device is in the
2763 process of canceling or aborting the job.
2764
2765 processingStopped(6),
2766 The job has stopped while processing for any number of
2767 reasons and will return to the processing state as soon as
2768 the reasons are no longer present.
2769
2770 The job's jmJobStateReasons1 object and/or the job's
2771 jobStateReasonsN (N=2..4) attributes MAY indicate why the
2772 job has stopped processing. For example, if the output
2773 device is stopped, the deviceStopped value MAY be included
2774 in the job's jmJobStateReasons1 object.
2775
2776 NOTE - When an output device is stopped, the device usually
2777 indicates its condition in human readable form at the
2778 device. The management application can obtain more
2779 complete device status remotely by querying the appropriate

2780 device MIB using the job's deviceIndex attribute(s), if the
2781 agent implements such a device MIB
2782
2783 canceled(7),
2784 A client has canceled the job and the server or device has
2785 completed canceling the job AND all MIB objects and
2786 attributes have reached their final values for the job.
2787 While the server or device is canceling the job, the job's
2788 jmJobStateReasons1 object SHOULD contain the
2789 processingToStopPoint value and one of the canceledByUser,
2790 canceledByOperator, or canceledAtDevice values. The
2791 canceledByUser, canceledByOperator, or canceledAtDevice
2792 values remain while the job is in the canceled state.
2793
2794 aborted(8),
2795 The job has been aborted by the system, usually while the
2796 job was in the processing or processingStopped state and
2797 the server or device has completed aborting the job AND all
2798 MIB objects and attributes have reached their final values
2799 for the job. While the server or device is aborting the
2800 job, the job's jmJobStateReasons1 object MAY contain the
2801 processingToStopPoint and abortedBySystem values. If
2802 implemented, the abortedBySystem value SHALL remain while
2803 the job is in the aborted state.
2804
2805 completed(9)
2806 The job has completed successfully or with warnings or
2807 errors after processing and all of the media have been
2808 successfully stacked in the appropriate output bin(s) AND
2809 all MIB objects and attributes have reached their final
2810 values for the job. The job's jmJobStateReasons1 object
2811 SHOULD contain one of: completedSuccessfully,
2812 completedWithWarnings, or completedWithErrors values."
2813 REFERENCE
2814 "
2815
2816 This is a type 2 enumeration. See Section 3.7.1.2."
2817 SYNTAX INTEGER {
2818 unknown(2),
2819 pending(3),
2820 pendingHeld(4),
2821 processing(5),
2822 processingStopped(6),
2823 canceled(7),
2824 aborted(8),
2825 completed(9)
2826 }

```

2827
2828 JmAttributeTypeTC ::= TEXTUAL-CONVENTION
2829     STATUS      current
2830     DESCRIPTION
2831         "The type of the attribute which identifies the attribute.
2832
2833         In the following definitions of the enums, each description
2834         indicates whether the useful value of the attribute SHALL be
2835         represented using the jmAttributeValueAsInteger or the
2836         jmAttributeValueAsOetets objects by the initial tag: 'INTEGER:'
2837         or 'OCTETS:', respectively.
2838
2839         Some attributes allow the agent implementer a choice of useful
2840         values of either an integer, an oetets representation, or both,
2841         depending on implementation. These attributes are indicated
2842         with 'INTEGER:' AND/OR 'OCTETS:' tags.
2843
2844         A very few attributes require both objects at the same time to
2845         represent a pair of useful values (see mediumConsumed(171)).
2846         These attributes are indicated with 'INTEGER:' AND 'OCTETS:'
2847         tags. See the jmAttributeGroup for the descriptions of these
2848         two MANDATORY objects.
2849
2850     NOTE - The enum assignments are grouped logically with values
2851     assigned in groups of 20, so that additional values may be
2852     registered in the future and assigned a value that is part of
2853     their logical grouping.
2854
2855     Values in the range 2**30 to 2**31-1 are reserved for private
2856     or experimental usage. This range corresponds to the same
2857     range reserved in IPP. Implementers are warned that use of
2858     such values may conflict with other implementations.
2859     Implementers are encouraged to request registration of enum
2860     values following the procedures in Section 3.7.1.
2861
2862     ─
2863
2864     REFERENCE
2865     ─See Section 3.2 entitled 'The Attribute Mechanism' for a
2866     description of this textual-convention and its use in the
2867     jmAttributeTable. See Section 3.3.8 for the specification of
2868     each attribute. The comment(s) after each enum assignment
2869     specifies the data type(s) of the attribute.
2870
2871     This is a type 2 enumeration. See Section 3.7.1.2."
2872
2873     SYNTAX      INTEGER {
2874         other(1),          -- Integer32 (-2..2147483647)
2875                           -- AND/OR
2876                           -- OCTET STRING(SIZE(0..63))
2877
2878     -- Job State attributes:

```

```
2879     jobStateReasons2(3),           -- JmJobStateReasons2TC
2880     jobStateReasons3(4),           -- JmJobStateReasons3TC
2881     jobStateReasons4(5),           -- JmJobStateReasons4TC
2882     processingMessage(6),           -- JmUTF8StringTC (SIZE(0..63))
2883     processingMessageNaturalLangTag(7),
2884                                     -- OCTET STRING(SIZE(0..63))
2885     jobCodedCharSet(8),             -- CodedCharSet
2886     jobNaturalLanguageTag(9),       -- OCTET STRING(SIZE(0..63))
2887
```

```

2888      -- Job Identification attributes:
2889      jobURI(20),                -- OCTET STRING(SIZE(0..63))
2890      jobAccountName(21),       -- OCTET STRING(SIZE(0..63))
2891      serverAssignedJobName(22), -- JmJobStringTC (SIZE(0..63))
2892      jobName(23),              -- JmJobStringTC (SIZE(0..63))
2893      jobServiceTypes(24),      -- JmJobServiceTypesTC
2894      jobSourceChannelIndex(25), -- Integer32 (0..2147483647)
2895      jobSourcePlatformType(26), -- JmJobSourcePlatformTypeTC
2896      submittingServerName(27),  -- JmJobStringTC (SIZE(0..63))
2897      submittingApplicationName(28), -- JmJobStringTC (SIZE(0..63))
2898      jobOriginatingHost(29),    -- JmJobStringTC (SIZE(0..63))
2899      deviceNameRequested(30),    -- JmJobStringTC (SIZE(0..63))
2900      queueNameRequested(31),    -- JmJobStringTC (SIZE(0..63))
2901      physicalDevice(32),        -- hrDeviceIndex
2902                                -- AND/OR
2903                                -- JmUTF8StringTC (SIZE(0..63))
2904      numberOfDocuments(33),     -- Integer32 (-2..2147483647)
2905      fileName(34),             -- JmJobStringTC (SIZE(0..63))
2906      documentName(35),         -- JmJobStringTC (SIZE(0..63))
2907      jobComment(36),           -- JmJobStringTC (SIZE(0..63))
2908      documentFormatIndex(37),   -- Integer32 (0..2147483647)
2909      documentFormat(38),       -- PrtInterpreterLangFamilyTC
2910                                -- AND/OR
2911                                -- OCTET STRING(SIZE(0..63))
2912
2913      -- Job Parameter attributes:
2914      jobPriority(50),           -- Integer32 (-2..100)
2915      jobProcessAfterDateAndTime(51), -- DateAndTime (SNMPv2-TC)
2916      jobHold(52),              -- JmBooleanTC
2917      jobHoldUntil(53),         -- JmJobStringTC (SIZE(0..63))
2918      outputBin(54),            -- Integer32 (0..2147483647)
2919                                -- AND/OR
2920                                -- JmJobStringTC (SIZE(0..63))
2921      sides(55),                -- Integer32 (-2..2)
2922      finishing(56),            -- JmFinishingTC
2923
2924      -- Image Quality attributes:
2925      printQualityRequested(70),  -- JmPrintQualityTC
2926      printQualityUsed(71),      -- JmPrintQualityTC
2927      printerResolutionRequested(72), -- JmPrinterResolutionTC
2928      printerResolutionUsed(73),   -- JmPrinterResolutionTC
2929      tonerEcomonyRequested(74),   -- JmTonerEconomyTC
2930      tonerEcomonyUsed(75),       -- JmTonerEconomyTC
2931      tonerDensityRequested(76),  -- Integer32 (-2..100)
2932      tonerDensityUsed(77),      -- Integer32 (-2..100)
2933

```

```

2934 -- Job Progress attributes:
2935 jobCopiesRequested(90), -- Integer32 (-2..2147483647)
2936 jobCopiesCompleted(91), -- Integer32 (-2..2147483647)
2937 documentCopiesRequested(92), -- Integer32 (-2..2147483647)
2938 documentCopiesCompleted(93), -- Integer32 (-2..2147483647)
2939 jobKOctetsTransferred(94), -- Integer32 (-2..2147483647)
2940 sheetCompletedCopyNumber(95), -- Integer32 (-2..2147483647)
2941 sheetCompletedDocumentNumber(96),
2942 -- Integer32 (-2..2147483647)
2943 jobCollationType(97), -- JmJobCollationTypeTC
2944
2945 -- Impression attributes:
2946 impressionsSpooled(110), -- Integer32 (-2..2147483647)
2947 impressionsSentToDevice(111), -- Integer32 (-2..2147483647)
2948 impressionsInterpreted(112), -- Integer32 (-2..2147483647)
2949 impressionsCompletedCurrentCopy(113),
2950 -- Integer32 (-2..2147483647)
2951 fullColorImpressionsCompleted(114),
2952 -- Integer32 (-2..2147483647)
2953 highlightColorImpressionsCompleted(115),
2954 -- Integer32 (-2..2147483647)
2955
2956 -- Page attributes:
2957 pagesRequested(130), -- Integer32 (-2..2147483647)
2958 pagesCompleted(131), -- Integer32 (-2..2147483647)
2959 pagesCompletedCurrentCopy(132), -- Integer32 (-2..2147483647)
2960
2961 -- Sheet attributes:
2962 sheetsRequested(150), -- Integer32 (-2..2147483647)
2963 sheetsCompleted(151), -- Integer32 (-2..2147483647)
2964 sheetsCompletedCurrentCopy(152), -- Integer32 (-2..2147483647)
2965
2966 -- Resource attributes:
2967 mediumRequested(170), -- JmMediumTypeTC
2968 -- AND/OR
2969 -- JmJobStringTC (SIZE(0..63))
2970 mediumConsumed(171), -- Integer32 (-2..2147483647)
2971 -- AND
2972 -- JmJobStringTC (SIZE(0..63))
2973 colorantRequested(172), -- Integer32 (-2..2147483647)
2974 -- AND/OR
2975 -- JmJobStringTC (SIZE(0..63))
2976 colorantConsumed(173), -- Integer32 (-2..2147483647)
2977 -- AND/OR
2978 -- JmJobStringTC (SIZE(0..63))
2979 mediumTypeConsumed(174), -- Integer32 (-2..2147483647)
2980 -- AND
2981 -- JmJobStringTC (SIZE(0..63))
2982 mediumSizeConsumed(175), -- Integer32 (-2..2147483647)
2983 -- AND
2984 -- JmJobStringTC (SIZE(0..63))
2985

```

```
2986      -- Time attributes:
2987      jobSubmissionToServerTime(190), -- JmTimeStampTC
2988                                     -- AND/OR
2989                                     -- DateAndTime
2990      jobSubmissionTime(191),        -- JmTimeStampTC
2991                                     -- AND/OR
2992                                     -- DateAndTime
2993      jobStartedBeingHeldTime(192),  -- JmTimeStampTC
2994                                     -- AND/OR
2995                                     -- DateAndTime
2996      jobStartedProcessingTime(193), -- JmTimeStampTC
2997                                     -- AND/OR
2998                                     -- DateAndTime
2999      jobCompletionTime(194),        -- JmTimeStampTC
3000                                     -- AND/OR
3001                                     -- DateAndTime
3002      jobProcessingCPUTime(195)     -- Integer32 (-2..2147483647)
3003  }
3004
```

3005 JmJobServiceTypesTC ::= TEXTUAL-CONVENTION
3006 STATUS current
3007 DESCRIPTION
3008 "Specifies the type(s) of service to which the job has been
3009 submitted (print, fax, scan, etc.). The service type is
3010 represented as an enum that is bit encoded with each job
3011 service type so that more general and arbitrary services can be
3012 created, such as services with more than one destination type,
3013 or ones with only a source or only a destination. For example,
3014 a job service might scan, faxOut, and print a single job. In
3015 this case, three bits would be set in the jobServiceTypes
3016 attribute, corresponding to the hexadecimal values: 0x8 + 0x20
3017 + 0x4, respectively, yielding: 0x2C.
3018
3019 Whether this attribute is set from a job attribute supplied by
3020 the job submission client or is set by the recipient job
3021 submission server or device depends on the job submission
3022 protocol. With either implementation, the agent SHALL return a
3023 non-zero value for this attribute indicating the type of the
3024 job.
3025
3026 One of the purposes of this attribute is to permit a requester
3027 to filter out jobs that are not of interest. For example, a
3028 printer operator MAY only be interested in jobs that include
3029 printing. That is why the attribute is in the job
3030 identification category.
3031
3032 The following service component types are defined (in
3033 hexadecimal) and are assigned a separate bit value for use with
3034 the jobServiceTypes attribute:
3035
3036 other 0x1
3037 The job contains some instructions that are not one of the
3038 identified types.
3039
3040 unknown 0x2
3041 The job contains some instructions whose type is unknown to
3042 the agent.
3043
3044 print 0x4
3045 The job contains some instructions that specify printing
3046
3047 scan 0x8
3048 The job contains some instructions that specify scanning
3049
3050 faxIn 0x10
3051 The job contains some instructions that specify receive fax
3052
3053 faxOut 0x20
3054 The job contains some instructions that specify sending fax
3055

3108 'jobIncoming' value and ending with the
3109 'jobCompletedWithErrors' value.
3110
3111 other 0x1
3112 The job state reason is not one of the standardized or
3113 registered reasons.
3114
3115 unknown 0x2
3116 The job state reason is not known to the agent or is
3117 indeterminent.
3118
3119 jobIncoming 0x4
3120 The job has been accepted by the server or device, but the
3121 server or device is expecting (1) additional operations
3122 from the client to finish creating the job and/or (2) is
3123 accessing/accepting document data.
3124
3125 submissionInterrupted 0x8
3126 The job was not completely submitted for some unforeseen
3127 reason, such as: (1) the server has crashed before the job
3128 was closed by the client, (2) the server or the document
3129 transfer method has crashed in some non-recoverable way
3130 before the document data was entirely transferred to the
3131 server, (3) the client crashed or failed to close the job
3132 before the time-out period.
3133
3134 jobOutgoing 0x10
3135 Configuration 2 only: The server is transmitting the job
3136 to the device.
3137
3138 jobHoldSpecified 0x20
3139 The value of the job's jobHold(52) attribute is TRUE. The
3140 job SHALL NOT be a candidate for processing until this
3141 reason is removed and there are no other reasons to hold
3142 the job.
3143
3144 jobHoldUntilSpecified 0x40
3145 The value of the job's jobHoldUntil(53) attribute specifies
3146 a time period that is still in the future. The job SHALL
3147 NOT be a candidate for processing until this reason is
3148 removed and there are no other reasons to hold the job.
3149
3150 jobProcessAfterSpecified 0x80
3151 The value of the job's jobProcessAfterDateAndTime(51)
3152 attribute specifies a time that is still in the future.
3153 The job SHALL NOT be a candidate for processing until this
3154 reason is removed and there are no other reasons to hold
3155 the job.
3156

3157 resourcesAreNotReady 0x100
3158 At least one of the resources needed by the job, such as
3159 media, fonts, resource objects, etc., is not ready on any
3160 of the physical devices for which the job is a candidate.
3161 This condition MAY be detected when the job is accepted, or
3162 subsequently while the job is pending or processing,
3163 depending on implementation.
3164

3165 deviceStoppedPartly 0x200
3166 One or more, but not all, of the devices to which the job
3167 is assigned are stopped. If all of the devices are stopped
3168 (or the only device is stopped), the deviceStopped reason
3169 SHALL be used.
3170

3171 deviceStopped 0x400
3172 The device(s) to which the job is assigned is (are all)
3173 stopped.
3174

3175 jobInterpreting 0x800
3176 The device to which the job is assigned is interpreting the
3177 document data.
3178

3179 jobPrinting 0x1000
3180 The output device to which the job is assigned is marking
3181 media. This value is useful for servers and output devices
3182 which spend a great deal of time processing (1) when no
3183 marking is happening and then want to show that marking is
3184 now happening or (2) when the job is in the process of
3185 being canceled or aborted while the job remains in the
3186 processing state, but the marking has not yet stopped so
3187 that impression or sheet counts are still increasing for
3188 the job.
3189

3190 jobCanceledByUser 0x2000
3191 The job was canceled by the owner of the job, i.e., by a
3192 user whose name is the same as the value of the job's
3193 jmJobOwner object, or by some other authorized end-user,
3194 such as a member of the job owner's security group.
3195

3196 jobCanceledByOperator 0x4000
3197 The job was canceled by the operator, i.e., by a user who
3198 has been authenticated as having operator privileges
3199 (whether local or remote).
3200

3201 jobCanceledAtDevice 0x8000
3202 The job was canceled by an unidentified local user, i.e., a
3203 user at a console at the device.
3204

3205 abortedBySystem 0x10000
3206 The job (1) is in the process of being aborted, (2) has
3207 been aborted by the system and placed in the 'aborted'
3208 state, or (3) has been aborted by the system and placed in
3209 the 'pendingHeld' state, so that a user or operator can
3210 manually try the job again.
3211

3212 processingToStopPoint 0x20000
3213 The requester has issued an operation to cancel or
3214 interrupt the job or the server/device has aborted the job,
3215 but the server/device is still performing some actions on
3216 the job until a specified stop point occurs or job
3217 termination/cleanup is completed.
3218

3219 This reason is recommended to be used in conjunction with
3220 the processing job state to indicate that the server/device
3221 is still performing some actions on the job while the job
3222 remains in the processing state. After all the job's
3223 resources consumed counters have stopped incrementing, the
3224 server/device moves the job from the processing state to
3225 the canceled or aborted job states.
3226

3227 serviceOffLine 0x40000
3228 The service or document transform is off-line and accepting
3229 no jobs. All pending jobs are put into the pendingHeld
3230 state. This situation could be true if the service's or
3231 document transform's input is impaired or broken.
3232

3233 jobCompletedSuccessfully 0x80000
3234 The job completed successfully.
3235

3236 jobCompletedWithWarnings 0x100000
3237 The job completed with warnings.
3238

3239 jobCompletedWithErrors 0x200000
3240 The job completed with errors (and possibly warnings too).
3241
3242

3243 The following additional job state reasons have been added to
3244 represent job states that are in ISO DPA[iso-dpa] and other job
3245 submission protocols:
3246

3247 jobPaused 0x400000
3248 The job has been indefinitely suspended by a client issuing
3249 an operation to suspend the job so that other jobs may
3250 proceed using the same devices. The client MAY issue an
3251 operation to resume the paused job at any time, in which
3252 case the agent SHALL remove the jobPaused values from the
3253 job's jmJobStateReasons1 object and the job is eventually
3254 resumed at or near the point where the job was paused.
3255

```

3256     jobInterrupted                0x800000
3257         The job has been interrupted while processing by a client
3258         issuing an operation that specifies another job to be run
3259         instead of the current job.  The server or device will
3260         automatically resume the interrupted job when the
3261         interrupting job completes.
3262
3263     jobRetained                      0x1000000
3264         The job is being retained by the server or device with all
3265         of the job's document data (and submitted resources, such
3266         as fonts, logos, and forms, if any).  Thus a client could
3267         issue an operation to the server or device to either (1)
3268         re-do the job (or a copy of the job) on the same server or
3269         device or (2) resubmit the job to another server or device.
3270         When a client could no longer re-do/resubmit the job, such
3271         as after the document data has been discarded, the agent
3272         SHALL remove the jobRetained value from the
3273         jmJobStateReasons1 object."
3274     REFERENCE
3275     
3276
3277         These bit definitions are the equivalent of a type 2 enum
3278         except that combinations of bits may be used together.  See
3279         section 3.7.1.2.  The remaining bits are reserved for future
3280         standardization and/or registration."
3281     SYNTAX      INTEGER (0..2147483647)    -- 31 bits, all but sign bit
3282
3283
3284
3285     JmJobStateReasons2TC ::= TEXTUAL-CONVENTION
3286         STATUS      current
3287         DESCRIPTION
3288             "This textual-convention is used with the jobStateReasons2
3289             attribute to provides additional information regarding the
3290             jmJobState object.  See the description under
3291             JmJobStateReasons1TC for additional information that applies to
3292             all reasons.
3293
3294             The following standard values are defined (in hexadecimal) as
3295             powers of two, since multiple values may be used at the same
3296             time:
3297
3298             cascaded                0x1
3299                 An outbound gateway has transmitted all of the job's job
3300                 and document attributes and data to another spooling
3301                 system.
3302
3303             deletedByAdministrator  0x2
3304                 The administrator has deleted the job.
3305
3306             discardTimeArrived      0x4
3307                 The job has been deleted due to the fact that the time

```

3308 specified by the job's job-discard-time attribute has
3309 arrived.
3310

3311 postProcessingFailed 0x8
3312 The post-processing agent failed while trying to log
3313 accounting attributes for the job; therefore the job has
3314 been placed into the completed state with the jobRetained
3315 jmJobStateReasons1 object value for a system-defined period
3316 of time, so the administrator can examine it, resubmit it,
3317 etc.
3318

3319 jobTransforming 0x10
3320 The server/device is interpreting document data and
3321 producing another electronic representation.
3322

3323 maxJobFaultCountExceeded 0x20
3324 The job has faulted several times and has exceeded the
3325 administratively defined fault count limit.
3326

3327 devicesNeedAttentionTimeOut 0x40
3328 One or more document transforms that the job is using needs
3329 human intervention in order for the job to make progress,
3330 but the human intervention did not occur within the site-
3331 settable time-out value.
3332

3333 needsKeyOperatorTimeOut 0x80
3334 One or more devices or document transforms that the job is
3335 using need a specially trained operator (who may need a key
3336 to unlock the device and gain access) in order for the job
3337 to make progress, but the key operator intervention did not
3338 occur within the site-settable time-out value.
3339

3340 jobStartWaitTimeOut 0x100
3341 The server/device has stopped the job at the beginning of
3342 processing to await human action, such as installing a
3343 special cartridge or special non-standard media, but the
3344 job was not resumed within the site-settable time-out value
3345 and the server/device has transitioned the job to the
3346 pendingHeld state.
3347

3348 jobEndWaitTimeOut 0x200
3349 The server/device has stopped the job at the end of
3350 processing to await human action, such as removing a
3351 special cartridge or restoring standard media, but the job
3352 was not resumed within the site-settable time-out value and
3353 the server/device has transitioned the job to the completed
3354 state.
3355

3356 jobPasswordWaitTimeOut 0x400
3357 The server/device has stopped the job at the beginning of
3358 processing to await input of the job's password, but the

3359 password was not received within the site-settable time-out
3360 value.
3361
3362 deviceTimedOut 0x800
3363 A device that the job was using has not responded in a
3364 period specified by the device's site-settable attribute.
3365
3366 connectingToDeviceTimeOut 0x1000
3367 The server is attempting to connect to one or more devices
3368 which may be dial-up, polled, or queued, and so may be busy
3369 with traffic from other systems, but server was unable to
3370 connect to the device within the site-settable time-out
3371 value.
3372
3373 transferring 0x2000
3374 The job is being transferred to a down stream server or
3375 downstream device.
3376
3377 queuedInDevice 0x4000
3378 The server/device has queued the job in a down stream
3379 server or downstream device.
3380
3381 jobQueued 0x8000
3382 The server/device has queued the document data.
3383
3384 jobCleanup 0x10000
3385 The server/device is performing cleanup activity as part of
3386 ending normal processing.
3387
3388 jobPasswordWait 0x20000
3389 The server/device has selected the job to be next to
3390 process, but instead of assigning resources and starting
3391 the job processing, the server/device has transitioned the
3392 job to the pendingHeld state to await entry of a password
3393 (and dispatched another job, if there is one).
3394
3395 validating 0x40000
3396 The server/device is validating the job *after* accepting the
3397 job.
3398
3399 queueHeld 0x80000
3400 The operator has held the entire job set or queue.
3401
3402 jobProofWait 0x100000
3403 The job has produced a single proof copy and is in the
3404 pendingHeld state waiting for the requester to issue an
3405 operation to release the job to print normally, obeying any
3406 job and document copy attributes that were originally
3407 submitted.
3408

3412 noSpaceOnServer 0x800000
3413 There is no room on the server to store all of the job.
3414
3415 pinRequired 0x1000000
3416 The System Administrator settable device policy is (1) to
3417 require PINs, and (2) to hold jobs that do not have a pin
3418 supplied as an input parameter when the job was created.
3419
3420 exceededAccountLimit 0x2000000
3421 The account for which this job is drawn has exceeded its
3422 limit. This condition SHOULD be detected before the job is
3423 scheduled so that the user does not wait until his/her job
3424 is scheduled only to find that the account is overdrawn.
3425 This condition MAY also occur while the job is processing
3426 either as processing begins or part way through processing.
3427
3428 heldForRetry 0x4000000
3429 The job encountered some errors that the server/device
3430 could not recover from with its normal retry procedures,
3431 but the error might not be encountered if the job is
3432 processed again in the future. Example cases are phone
3433 number busy or remote file system in-accessible. For such
3434 a situation, the server/device SHALL transition the job
3435 from the processing to the pendingHeld, rather than to the
3436 aborted state.
3437
3438 The following values are from the X/Open PSIS draft standard:
3439
3440 canceledByShutdown 0x8000000
3441 The job was canceled because the server or device was
3442 shutdown before completing the job.
3443
3444 deviceUnavailable 0x10000000
3445 This job was aborted by the system because the device is
3446 currently unable to accept jobs.
3447
3448 wrongDevice 0x20000000
3449 This job was aborted by the system because the device is
3450 unable to handle this particular job; the spooler SHOULD
3451 try another device or the user should submit the job to
3452 another device.
3453
3454 badJob 0x40000000
3455 This job was aborted by the system because this job has a
3456 major problem, such as an ill-formed PDL; the spooler
3457 SHOULD not even try another device.
3458 **REFERENCE**
3459
3460
3461 These bit definitions are the equivalent of a type 2 enum
3462 except that combinations of them may be used together. See

3463 section 3.7.1.2. See the description under
3464 JmJobStateReasons1TC and the jobStateReasons2 attribute."
3465 SYNTAX INTEGER (0..2147483647) -- 31 bits, all but sign bit
3466
3467 JmJobStateReasons3TC ::= TEXTUAL-CONVENTION
3468 STATUS current
3469 DESCRIPTION
3470 "This textual-convention is used with the jobStateReasons3
3471 attribute to provides additional information regarding the
3472 jmJobState object. See the description under
3473 JmJobStateReasons1TC for additional information that applies to
3474 all reasons.
3475
3476 The following standard values are defined (in hexadecimal) as
3477 *powers of two*, since multiple values may be used at the same
3478 time:
3479
3480 jobInterruptedByDeviceFailure 0x1
3481 A device or the print system software that the job was
3482 using has failed while the job was processing. The server
3483 or device is keeping the job in the pendingHeld state until
3484 an operator can determine what to do with the job."
3485 REFERENCE
3486 "
3487
3488 These bit definitions are the equivalent of a type 2 enum
3489 except that combinations of them may be used together. See
3490 section 3.7.1.2. The remaining bits are reserved for future
3491 standardization and/or registration. See the description under
3492 JmJobStateReasons1TC and the jobStateReasons3 attribute."
3493 SYNTAX INTEGER (0..2147483647) -- 31 bits, all but sign bit
3494
3495
3496
3497
3498
3499 JmJobStateReasons4TC ::= TEXTUAL-CONVENTION
3500 STATUS current
3501 DESCRIPTION
3502 "This textual-convention is used in the jobStateReasons4
3503 attribute to provides additional information regarding the
3504 jmJobState object. See the description under
3505 JmJobStateReasons1TC for additional information that applies to
3506 all reasons.
3507
3508 The following standard values are defined (in hexadecimal) as
3509 *powers of two*, since multiple values may be used at the same
3510 time:
3511
3512 none yet defined. These bits are reserved for future
3513 standardization and/or registration."
3514 REFERENCE

3515 "
3516
3517 These bit definitions are the equivalent of a type 2 enum
3518 except that combinations of them may be used together. See
3519 section 3.7.1.2. See the description under
3520 JmJobStateReasons1TC and the jobStateReasons4 attribute."
3521 SYNTAX INTEGER (0..2147483647) -- 31 bits, all but sign bit


```
3570 jmGeneralJobSetIndex OBJECT-TYPE
3571     SYNTAX      Integer32 (1..32767)
3572     MAX-ACCESS  not-accessible
3573     STATUS      current
3574     DESCRIPTION
3575         "A unique value for each job set in this MIB.  The jmJobTable
3576         and jmAttributeTable tables have this same index as their
3577         primary index.
3578
3579         The value(s) of the jmGeneralJobSetIndex SHALL be persistent
3580         across power cycles, so that clients that have retained
3581         jmGeneralJobSetIndex values will access the same job sets upon
3582         subsequent power-up.
3583
3584         An implementation that has only one job set, such as a printer
3585         with a single queue, SHALL hard code this object with the value
3586         1."
3587     REFERENCE
3588         "
3589
3590         See Section 2 entitled 'Terminology and Job Model' for the
3591         definition of a job set.
3592         Corresponds to the first index in jmJobTable and
3593         jmAttributeTable."
3594     ::= { jmGeneralEntry 1 }
3595
3596
3597 jmGeneralNumberOfActiveJobs OBJECT-TYPE
3598     SYNTAX      Integer32 (0..2147483647)
3599     MAX-ACCESS  read-only
3600     STATUS      current
3601     DESCRIPTION
3602         "The current number of 'active' jobs in the jmJobIDTable,
3603         jmJobTable, and jmAttributeTable, i.e., the total number of
3604         jobs that are in the pending, processing, or processingStopped
3605         states.  See the JmJobStateTC textual-convention for the exact
3606         specification of the semantics of the job states."
3607     DEFVAL      { 0 }      -- no jobs
3608     ::= { jmGeneralEntry 2 }
3609
```

```
3610 jmGeneralOldestActiveJobIndex OBJECT-TYPE
3611     SYNTAX      Integer32 (0..2147483647)
3612     MAX-ACCESS  read-only
3613     STATUS      current
3614     DESCRIPTION
3615         "The jmJobIndex of the oldest job that is still in one of the
3616         'active' states (pending, processing, or processingStopped).
3617         In other words, the index of the 'active' job that has been in
3618         the job tables the longest.
3619
3620         If there are no active jobs, the agent SHALL set the value of
3621         this object to 0."u
3622     REFERENCE
3623     u
3624
3625         See Section 3.2 entitled 'The Job Tables and the Oldest Active
3626         and Newest Active Indexes' for a description of the usage of
3627         this object."
3628     DEFVAL      { 0 }      -- no active jobs
3629     ::= { jmGeneralEntry 3 }
3630
3631
3632
3633 jmGeneralNewestActiveJobIndex OBJECT-TYPE
3634     SYNTAX      Integer32 (0..2147483647)
3635     MAX-ACCESS  read-only
3636     STATUS      current
3637     DESCRIPTION
3638         "The jmJobIndex of the newest job that is in one of the
3639         'active' states (pending, processing, or processingStopped).
3640         In other words, the index of the 'active' job that has been
3641         most recently added to the job tables.
3642
3643         When all jobs become 'inactive', i.e., enter the pendingHeld,
3644         completed, canceled, or aborted states, the agent SHALL set the
3645         value of this object to 0."u
3646     REFERENCE
3647     u
3648
3649         See Section 3.2 entitled 'The Job Tables and the Oldest Active
3650         and Newest Active Indexes' for a description of the usage of
3651         this object."
3652     DEFVAL      { 0 }      -- no active jobs
3653     ::= { jmGeneralEntry 4 }
3654
```

```
3655 jmGeneralJobPersistence OBJECT-TYPE
3656     SYNTAX      Integer32 (15..2147483647)
3657     UNITS       "seconds"
3658     MAX-ACCESS  read-only
3659     STATUS      current
3660     DESCRIPTION
3661         "The minimum time in seconds for this instance of the Job Set
3662         that an entry SHALL remain in the jmJobIDTable and jmJobTable
3663         after processing has completed, i.e., the minimum time in
3664         seconds starting when the job enters the completed, canceled,
3665         or aborted state.
3666
3667         Configuring this object is implementation-dependent.
3668
3669         This value SHALL be equal to or greater than the value of
3670         jmGeneralAttributePersistence. This value SHOULD be at least
3671         60 which gives a monitoring or accounting application one
3672         minute in which to poll for job data."
3673     DEFVAL      { 60 }          -- one minute
3674     ::= { jmGeneralEntry 5 }
3675
3676
3677
3678 jmGeneralAttributePersistence OBJECT-TYPE
3679     SYNTAX      Integer32 (15..2147483647)
3680     UNITS       "seconds"
3681     MAX-ACCESS  read-only
3682     STATUS      current
3683     DESCRIPTION
3684         "The minimum time in seconds for this instance of the Job Set
3685         that an entry SHALL remain in the jmAttributeTable after
3686         processing has completed , i.e., the time in seconds starting
3687         when the job enters the completed, canceled, or aborted state.
3688
3689         Configuring this object is implementation-dependent.
3690
3691         This value SHOULD be at least 60 which gives a monitoring or
3692         accounting application one minute in which to poll for job
3693         data."
3694     DEFVAL      { 60 }          -- one minute
3695     ::= { jmGeneralEntry 6 }
3696
```

```
3697 jmGeneralJobSetName OBJECT-TYPE
3698     SYNTAX      JmUTF8StringTC (SIZE(0..63))
3699     MAX-ACCESS  read-only
3700     STATUS      current
3701     DESCRIPTION
3702         "The human readable name of this job set assigned by the system
3703         administrator (by means outside of this MIB). Typically, this
3704         name SHOULD be the name of the job queue. If a server or
3705         device has only a single job set, this object can be the
3706         administratively assigned name of the server or device itself.
3707         This name does not need to be unique, though each job set in a
3708         single Job Monitoring MIB SHOULD have distinct names.
3709
3710         NOTE - If the job set corresponds to a single printer and the
3711         Printer MIB is implemented, this value SHOULD be the same as
3712         the prtGeneralPrinterName object in the draft Printer MIB
3713         [print-mib-draft]. If the job set corresponds to an IPP
3714         Printer, this value SHOULD be the same as the IPP 'printer-
3715         name' Printer attribute.
3716
3717         NOTE - The purpose of this object is to help the user of the
3718         job monitoring application distinguish between several job sets
3719         in implementations that support more than one job set."
3720     REFERENCE
3721         
3722
3723         See the OBJECT compliance macro for the minimum maximum length
3724         required for conformance."
3725     DEFVAL      { 'H } -- empty string
3726     ::= { jmGeneralEntry 7 }
3727
3728
3729
3730
3731
```



```

3732 -- The Job ID Group (MANDATORY)
3733
3734 -- The jmJobIDGroup consists entirely of the jmJobIDTable.
3735
3736 jmJobID OBJECT IDENTIFIER ::= { jobmonMIBObjects 2 }
3737
3738 jmJobIDTable OBJECT-TYPE
3739     SYNTAX      SEQUENCE OF JmJobIDEntry
3740     MAX-ACCESS  not-accessible
3741     STATUS      current
3742     DESCRIPTION
3743         "The jmJobIDTable provides a correspondence map (1) between the
3744         job submission ID that a client uses to refer to a job and (2)
3745         the jmGeneralJobSetIndex and jmJobIndex that the Job Monitoring
3746         MIB agent assigned to the job and that are used to access the
3747         job in all of the other tables in the MIB.  If a monitoring
3748         application already knows the jmGeneralJobSetIndex and the
3749         jmJobIndex of the job it is querying, that application NEED NOT
3750         use the jmJobIDTable."
3751     REFERENCE
3752         
3753
3754         The MANDATORY-GROUP macro specifies that this group is
3755         MANDATORY."
3756     ::= { jmJobID 1 }
3757
3758
3759
3760 jmJobIDEntry OBJECT-TYPE
3761     SYNTAX      JmJobIDEntry
3762     MAX-ACCESS  not-accessible
3763     STATUS      current
3764     DESCRIPTION
3765         "The map from (1) the jmJobSubmissionID to (2) the
3766         jmGeneralJobSetIndex and jmJobIndex.
3767
3768         An entry SHALL exist in this table for each job currently known
3769         to the agent for all job sets and job states.  There MAY be
3770         more than one jmJobIDEntry that maps to a single job.  This
3771         many to one mapping can occur when more than one network entity
3772         along the job submission path supplies a job submission ID.
3773         See Section 3.5.  However, each job SHALL appear once and in
3774         one and only one job set."
3775     INDEX      { jmJobSubmissionID }
3776     ::= { jmJobIDTable 1 }
3777
3778 JmJobIDEntry ::= SEQUENCE {
3779     jmJobSubmissionID          OCTET STRING(SIZE(48)),
3780     jmJobIDJobSetIndex        Integer32 (0..32767),
3781     jmJobIDJobIndex           Integer32 (0..2147483647)
3782 }
3783

```

```
3784 jmJobSubmissionID OBJECT-TYPE
3785     SYNTAX      OCTET STRING(SIZE(48))
3786     MAX-ACCESS  not-accessible
3787     STATUS      current
3788     DESCRIPTION
3789         "A quasi-unique 48-octet fixed-length string ID which
3790         identifies the job within a particular client-server
3791         environment.  There are multiple formats for the
3792         jmJobSubmissionID.  Each format SHALL be uniquely identified.
3793         See the JmJobSubmissionIDTypeTC textual convention.  Each
3794         format SHALL be registered using the procedures of a type 2
3795         enum.  See section 3.7.3 entitled: 'PWG Registration of Job
3796         Submission Id Formats'.
3797
3798         If the requester (client or server) does not supply a job
3799         submission ID in the job submission protocol, then the
3800         recipient (server or device) SHALL assign a job submission ID
3801         using any of the standard formats that have been reserved for
3802         agents and adding the final 8 octets to distinguish the ID from
3803         others submitted from the same requester.
3804
3805         The monitoring application, whether in the client or running
3806         separately, MAY use the job submission ID to help identify
3807         which jmJobIndex was assigned by the agent, i.e., in which row
3808         the job information is in the other tables.
3809
3810         NOTE - fixed-length is used so that a management application
3811         can use a shortened GetNext varbind (in SNMPv1 and SNMPv2) in
3812         order to get the next submission ID, disregarding the remainder
3813         of the ID in order to access jobs independent of the trailing
3814         identifier part, e.g., to get all jobs submitted by a
3815         particular jmJobOwner or submitted from a particular MAC
3816         address."
3817     REFERENCE
3818         
3819
3820         See the JmJobSubmissionIDTypeTC textual convention.
3821         See APPENDIX B - Support of Job Submission Protocols."
3822     ::= { jmJobIDEntry 1 }
3823
```

```
3824 jmJobIDJobSetIndex OBJECT-TYPE
3825     SYNTAX      Integer32 (0..32767)
3826     MAX-ACCESS  read-only
3827     STATUS      current
3828     DESCRIPTION
3829         "This object contains the value of the jmGeneralJobSetIndex for
3830         the job with the jmJobSubmissionID value, i.e., the job set
3831         index of the job set in which the job was placed when that
3832         server or device accepted the job. This 16-bit value in
3833         combination with the jmJobIDJobIndex value permits the
3834         management application to access the other tables to obtain the
3835         job-specific objects for this job."
3836     REFERENCE
3837         
3838
3839         See jmGeneralJobSetIndex in the jmGeneralTable."
3840     DEFVAL      { 0 }      -- 0 indicates no job set index
3841     ::= { jmJobIDEntry 2 }
3842
3843
3844
3845 jmJobIDJobIndex OBJECT-TYPE
3846     SYNTAX      Integer32 (0..2147483647)
3847     MAX-ACCESS  read-only
3848     STATUS      current
3849     DESCRIPTION
3850         "This object contains the value of the jmJobIndex for the job
3851         with the jmJobSubmissionID value, i.e., the job index for the
3852         job when the server or device accepted the job. This value, in
3853         combination with the jmJobIDJobSetIndex value, permits the
3854         management application to access the other tables to obtain the
3855         job-specific objects for this job."
3856     REFERENCE
3857         
3858
3859         See jmJobIndex in the jmJobTable."
3860     DEFVAL      { 0 }      -- 0 indicates no jmJobIndex value.
3861     ::= { jmJobIDEntry 3 }
3862
3863
3864
3865
```

```

3866 -- The Job Group (MANDATORY)
3867
3868 -- The jmJobGroup consists entirely of the jmJobTable.
3869
3870 jmJob OBJECT IDENTIFIER ::= { jobmonMIBObjects 3 }
3871
3872 jmJobTable OBJECT-TYPE
3873     SYNTAX      SEQUENCE OF JmJobEntry
3874     MAX-ACCESS  not-accessible
3875     STATUS      current
3876     DESCRIPTION
3877         "The jmJobTable consists of basic job state and status
3878         information for each job in a job set that (1) monitoring
3879         applications need to be able to access in a single SNMP Get
3880         operation, (2) that have a single value per job, and (3) that
3881         SHALL always be implemented."
3882     REFERENCE
3883         
3884
3885         The MANDATORY-GROUP macro specifies that this group is
3886         MANDATORY."
3887     ::= { jmJob 1 }
3888
3889
3890
3891 jmJobEntry OBJECT-TYPE
3892     SYNTAX      JmJobEntry
3893     MAX-ACCESS  not-accessible
3894     STATUS      current
3895     DESCRIPTION
3896         "Basic per-job state and status information.
3897
3898         An entry SHALL exist in this table for each job, no matter what
3899         the state of the job is. Each job SHALL appear in one and only
3900         one job set."
3901     REFERENCE
3902         
3903
3904         See Section 3.2 entitled 'The Job Tables'."
3905     INDEX { jmGeneralJobSetIndex, jmJobIndex }
3906     ::= { jmJobTable 1 }
3907
3908 JmJobEntry ::= SEQUENCE {
3909     jmJobIndex          Integer32 (1..2147483647),
3910     jmJobState          JmJobStateTC,
3911     jmJobStateReasons1 JmJobStateReasons1TC,
3912     jmNumberOfInterveningJobs Integer32 (-2..2147483647),
3913     jmJobKOctetsPerCopyRequested Integer32 (-2..2147483647),
3914     jmJobKOctetsProcessed Integer32 (-2..2147483647),
3915     jmJobImpressionsPerCopyRequested Integer32 (-2..2147483647),
3916     jmJobImpressionsCompleted Integer32 (-2..2147483647),
3917     jmJobOwner          JmJobStringTC (SIZE(0..63))

```

3918 }
3919

```
3920 jmJobIndex OBJECT-TYPE
3921     SYNTAX      Integer32 (1..2147483647)
3922     MAX-ACCESS  not-accessible
3923     STATUS      current
3924     DESCRIPTION
3925         "The sequential, monotonically increasing identifier index for
3926         the job generated by the server or device when that server or
3927         device accepted the job. This index value permits the
3928         management application to access the other tables to obtain the
3929         job-specific row entries."u
3930     REFERENCE
3931         u
3932
3933         See Section 3.2 entitled 'The Job Tables and the Oldest Active
3934         and Newest Active Indexes'.
3935         See Section 3.5 entitled 'Job Identification'.
3936         See also
3937
3938         jmGeneralNewestActiveJobIndex for the largest value of
3939         jmJobIndex.
3940         See JmJobSubmissionIDTypeTC for a limit on the size of this
3941         index if the agent represents it as an 8-digit decimal number."
3942     ::= { jmJobEntry 1 }
3943
3944
3945
3946 jmJobState OBJECT-TYPE
3947     SYNTAX      JmJobStateTC
3948     MAX-ACCESS  read-only
3949     STATUS      current
3950     DESCRIPTION
3951         "The current state of the job (pending, processing, completed,
3952         etc.). Agents SHALL implement only those states which are
3953         appropriate for the particular implementation. However,
3954         management applications SHALL be prepared to receive all the
3955         standard job states.
3956
3957         The final value for this object SHALL be one of: completed,
3958         canceled, or aborted. The minimum length of time that the
3959         agent SHALL maintain MIB data for a job in the completed,
3960         canceled, or aborted state before removing the job data from
3961         the jmJobIDTable and jmJobTable is specified by the value of
3962         the jmGeneralJobPersistence object."
3963     DEFVAL      { unknown }          -- default is unknown
3964     ::= { jmJobEntry 2 }
3965
```

```
3966 jmJobStateReasons1 OBJECT-TYPE
3967     SYNTAX      JmJobStateReasons1TC
3968     MAX-ACCESS  read-only
3969     STATUS      current
3970     DESCRIPTION
3971         "Additional information about the job's current state, i.e.,
3972         information that augments the value of the job's jmJobState
3973         object.
3974
3975         Implementation of any reason values is OPTIONAL, but an agent
3976         SHOULD return any reason information available. These values
3977         MAY be used with any job state or states for which the reason
3978         makes sense. Since the Job State Reasons will be more dynamic
3979         than the Job State, it is recommended that a job monitoring
3980         application read this object every time jmJobState is read.
3981         When the agent cannot provide a reason for the current state of
3982         the job, the value of the jmJobStateReasons1 object and
3983         jobStateReasonsN attributes SHALL be 0.
3984         "
3985     REFERENCE
3986         "The jobStateReasonsN (N=2..4) attributes provide further
3987         additional information about the job's current state."
3988     DEFVAL      { 0 }          -- no reasons
3989     ::= { jmJobEntry 3 }
3990
3991
3992
3993 jmNumberOfInterveningJobs OBJECT-TYPE
3994     SYNTAX      Integer32 (-2..2147483647)
3995     MAX-ACCESS  read-only
3996     STATUS      current
3997     DESCRIPTION
3998         "The number of jobs that are expected to complete processing
3999         before this job has completed processing according to the
4000         implementation's queuing algorithm, if no other jobs were to be
4001         submitted. In other words, this value is the job's queue
4002         position. The agent SHALL return a value of 0 for this
4003         attribute when the job is the next job to complete processing
4004         (or has completed processing)."
4005     DEFVAL      { 0 }          -- default is no intervening jobs.
4006     ::= { jmJobEntry 4 }
4007
```

```
4008 jmJobKOctetsPerCopyRequested OBJECT-TYPE
4009     SYNTAX      Integer32 (-2..2147483647)
4010     MAX-ACCESS  read-only
4011     STATUS      current
4012     DESCRIPTION
4013         "The total size in K (1024) octets of the document(s) being
4014         requested to be processed in the job.  The agent SHALL round
4015         the actual number of octets up to the next highest K.  Thus 0
4016         octets SHALL beis represented as '0', 1-1024 octets SHALL beis
4017         represented as '1', 1025-2048 SHALL beis represented as '2',
4018         etc.
4019
4020         In computing this value, the server/device SHALL NOT not
4021         include the multiplicative factors contributed by (1) the
4022         number of document copies, and (2) the number of job copies,
4023         independent of whether the device can process multiple copies
4024         of the job or document without making multiple passes over the
4025         job or document data and independent of whether the output is
4026         collated or not.  Thus the server/device computation is
4027         independent of the implementation and indicates the size of the
4028         document(s) measured in K octets independent of the number of
4029         copies."
4030     DEFVAL      { -2 }      -- the default is unknown(-2)
4031     ::= { jmJobEntry 5 }
```

```
4032
4033
4034
4035 jmJobKOctetsProcessed OBJECT-TYPE
4036     SYNTAX      Integer32 (-2..2147483647)
4037     MAX-ACCESS  read-only
4038     STATUS      current
4039     DESCRIPTION
4040         "The total number of octets processed by the server or device
4041         measured in units of K (1024) octets so far.  The agent SHALL
4042         round the actual number of octets processed up to the next
4043         higher K.  Thus 0 octets SHALL beis represented as '0', 1-1024
4044         octets SHALL beis represented as '1', 1025-2048 octets SHALL
4045         beis '2', etc.  For printing devices, this value is the number
4046         interpreted by the page description language interpreter rather
4047         than what has been marked on media.
4048
4049         For implementations where multiple copies are produced by the
4050         interpreter with only a single pass over the data, the final
4051         value SHALL be equal to the value of the
4052         jmJobKOctetsPerCopyRequested object.  For implementations where
4053         multiple copies are produced by the interpreter by processing
4054         the data for each copy, the final value SHALL be a multiple of
4055         the value of the jmJobKOctetsPerCopyRequested object.
4056
4057         NOTE - See the impressionsCompletedCurrentCopy and
4058         pagesCompletedCurrentCopy attributes for attributes that are
4059         reset on each document copy.
```


4060
4061 NOTE - The jmJobKOctetsProcessed object can be used with the
4062 jmJobKOctetsPerCopyRequested object to provide an indication of
4063 the relative progress of the job, provided that the
4064 multiplicative factor is taken into account for some
4065 implementations of multiple copies."
4066 DEFVAL { 0 } -- default is no octets processed.
4067 ::= { jmJobEntry 6 }
4068
4069
4070 jmJobImpressionsPerCopyRequested OBJECT-TYPE
4071 SYNTAX Integer32 (-2..2147483647)
4072 MAX-ACCESS read-only
4073 STATUS current
4074 DESCRIPTION
4075 "The total size in number of impressions of the document(s)
4076 submitted.
4077
4078 In computing this value, the server/device ~~SHALL NOT~~ include
4079 the multiplicative factors contributed by (1) the number of
4080 document copies, and (2) the number of job copies, independent
4081 of whether the device can process multiple copies of the job or
4082 document without making multiple passes over the job or
4083 document data and independent of whether the output is collated
4084 or not. Thus the server/device computation is independent of
4085 the implementation and reflects the size of the document(s)
4086 measured in impressions independent of the number of copies."
4087 REFERENCE
4088 "
4089
4090 See the definition of the term 'impression' in Section 2."
4091 DEFVAL { -2 } -- default is unknown(-2)
4092 ::= { jmJobEntry 7 }
4093
4094
4095 jmJobImpressionsCompleted OBJECT-TYPE
4096 SYNTAX Integer32 (-2..2147483647)
4097 MAX-ACCESS read-only
4098 STATUS current
4099 DESCRIPTION
4100 "The total number of impressions completed for this job so far.
4101 For printing devices, the impressions completed includes
4102 interpreting, marking, and stacking the output. For other
4103 types of job services, the number of impressions completed
4104 includes the number of impressions processed.
4105
4106 NOTE - See the impressionsCompletedCurrentCopy and
4107 pagesCompletedCurrentCopy attributes for attributes that are
4108 reset on each document copy.
4109
4110 NOTE - The jmJobImpressionsCompleted object can be used with
4111 the jmJobImpressionsPerCopyRequested object to provide an

4112 indication of the relative progress of the job, provided that
4113 the multiplicative factor is taken into account for some
4114 implementations of multiple copies."
4115 REFERENCE
4116 "
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 REFERENCE
4140 "
4141
4142 See the OBJECT compliance macro for the minimum maximum length
4143 required for conformance."
4144 DEFVAL { ''H } -- default is empty string
4145 ::= { jmJobEntry 9 }
4146
4147
4148
4149

```
4150 -- The Attribute Group (MANDATORY)
4151
4152 -- The jmAttributeGroup consists entirely of the jmAttributeTable.
4153 --
4154 -- Implementation of the objects in this group is MANDATORY.
4155 -- See Section 3.1 entitled 'Conformance Considerations'.
4156 -- An agent SHALL implement any attribute if (1) the server or device
4157 -- supports the functionality represented by the attribute and (2) the
4158 -- information is available to the agent.
4159
4160 jmAttribute OBJECT IDENTIFIER ::= { jobmonMIBObjects 4 }
4161
4162
4163
4164 jmAttributeTable OBJECT-TYPE
4165     SYNTAX          SEQUENCE OF JmAttributeEntry
4166     MAX-ACCESS      not-accessible
4167     STATUS          current
4168     DESCRIPTION
4169         "The jmAttributeTable SHALL contain attributes of the job and
4170         document(s) for each job in a job set.  Instead of allocating
4171         distinct objects for each attribute, each attribute is
4172         represented as a separate row in the jmAttributeTable."u
4173     REFERENCE
4174     u
4175
4176         The MANDATORY-GROUP macro specifies that this group is
4177         MANDATORY.  An agent SHALL implement any attribute if (1) the
4178         server or device supports the functionality represented by the
4179         attribute and (2) the information is available to the agent. "
4180     ::= { jmAttribute 1 }
4181
4182
4183
4184 jmAttributeEntry OBJECT-TYPE
4185     SYNTAX          JmAttributeEntry
4186     MAX-ACCESS      not-accessible
4187     STATUS          current
4188     DESCRIPTION
4189         "Attributes representing information about the job and
4190         document(s) or resources required and/or consumed.
4191
4192         Each entry in the jmAttributeTable is a per-job entry with an
4193         extra index for each type of attribute (jmAttributeTypeIndex)
4194         that a job can have and an additional index
4195         (jmAttributeInstanceIndex) for those attributes that can have
4196         multiple instances per job.  The jmAttributeTypeIndex object
4197         SHALL contain an enum type that indicates the type of attribute
4198         (see the JmAttributeTypeTC textual-convention).  The value of
4199         the attribute SHALL be represented in either the
4200         jmAttributeValueAsInteger or jmAttributeValueAsOctets objects,
```

4201 and/or both, as specified in the JmAttributeTypeTC textual-
4202 convention.
4203

4204 The agent SHALL create rows in the jmAttributeTable as the
4205 server or device is able to discover the attributes either from
4206 the job submission protocol itself or from the document PDL.
4207 As the documents are interpreted, the interpreter MAY discover
4208 additional attributes and so the agent adds additional rows to
4209 this table. As the attributes that represent resources are
4210 actually consumed, the usage counter contained in the
4211 jmAttributeValueAsInteger object is incremented according to
4212 the units indicated in the description of the JmAttributeTypeTC
4213 enum.
4214

4215 The agent SHALL maintain each row in the ~~jmAttributeJob~~Table
4216 for at least the minimum time after a job completes as
4217 specified by the jmGeneralAttributePersistence object.
4218

4219 Zero or more entries SHALL exist in this table for each job in
4220 a job set."
4221 **REFERENCE**
4222 "
4223

4224 See Section 3.3 entitled 'The Attribute Mechanism' for a
4225 description of the jmAttributeTable."
4226 INDEX { jmGeneralJobSetIndex, jmJobIndex, jmAttributeTypeIndex,
4227 jmAttributeInstanceIndex }
4228 ::= { jmAttributeTable 1 }
4229

4230 JmAttributeEntry ::= SEQUENCE {
4231 jmAttributeTypeIndex JmAttributeTypeTC,
4232 jmAttributeInstanceIndex Integer32 (1..32767),
4233 jmAttributeValueAsInteger Integer32 (-2..2147483647),
4234 jmAttributeValueAsOctets OCTET STRING(SIZE(0..63))
4235 }
4236

```
4237 jmAttributeTypeIndex OBJECT-TYPE
4238     SYNTAX          JmAttributeTypeTC
4239     MAX-ACCESS     not-accessible
4240     STATUS         current
4241     DESCRIPTION
4242         "The type of attribute that this row entry represents.
4243
4244         The type MAY identify information about the job or document(s)
4245         or MAY identify a resource required to process the job before
4246         the job start processing and/or consumed by the job as the job
4247         is processed.
4248
4249         Examples of job attributes (i.e., apply to the job as a whole)
4250         that have only one instance per job include:
4251         jobCopiesRequested(90), documentCopiesRequested(92),
4252         jobCopiesCompleted(91), documentCopiesCompleted(93), while
4253         examples of job attributes that may have more than one instance
4254         per job include: documentFormatIndex(37), and
4255         documentFormat(38).
4256
4257         Examples of document attributes (one instance per document)
4258         include: fileName(34), and documentName(35).
4259
4260         Examples of required and consumed resource attributes include:
4261         pagesRequested(130), mediumRequested(170), pagesCompleted(131),
4262         and mediumConsumed(171), respectively."
4263     ::= { jmAttributeEntry 1 }
4264
4265
4266
4267 jmAttributeInstanceIndex OBJECT-TYPE
4268     SYNTAX          Integer32 (1..32767)
4269     MAX-ACCESS     not-accessible
4270     STATUS         current
4271     DESCRIPTION
4272         "A running 16-bit index of the attributes of the same type for
4273         each job.  For those attributes with only a single instance per
4274         job, this index value SHALL be 1.  For those attributes that
4275         are a single value per document, the index value SHALL be the
4276         document number, starting with 1 for the first document in the
4277         job.  Jobs with only a single document SHALL use the index
4278         value of 1.  For those attributes that can have multiple values
4279         per job or per document, such as documentFormatIndex(37) or
4280         documentFormat(38), the index SHALL be a running index for the
4281         job as a whole, starting at 1."
4282     ::= { jmAttributeEntry 2 }
4283
```

```
4284 jmAttributeValueAsInteger OBJECT-TYPE
4285     SYNTAX      Integer32 (-2..2147483647)
4286     MAX-ACCESS  read-only
4287     STATUS      current
4288     DESCRIPTION
4289         "The integer value of the attribute.  The value of the
4290         attribute SHALL be represented as an integer if the enum
4291         description in the JmAttributeTypeTC textual-convention
4292         definition has the tag: 'INTEGER:'.
```

4293

4294 Depending on the enum definition, this object value MAY be an
4295 integer, a counter, an index, or an enum, depending on the
4296 jmAttributeTypeIndex value. The units of this value are
4297 specified in the enum description.

4298

4299 For those attributes that are accumulating job consumption as
4300 the job is processed as specified in the JmAttributeTypeTC
4301 textual-convention, SHALL contain the final value after the job
4302 completes processing, i.e., this value SHALL indicate the total
4303 usage of this resource made by the job.

4304

4305 A monitoring application is able to copy this value to a
4306 suitable longer term storage for later processing as part of an
4307 accounting system.

4308

4309 Since the agent MAY add attributes representing resources to
4310 this table while the job is waiting to be processed or being
4311 processed, which can be a long time before any of the resources
4312 are actually used, the agent SHALL set the value of the
4313 jmAttributeValueAsInteger object to 0 for resources that the
4314 job has not yet consumed.

4315

4316 Attributes for which the concept of an integer value is
4317 meaningless, such as fileName(34), jobName, and
4318 processingMessage, do not have the 'INTEGER:' tag in the
4319 JmAttributeTypeTC definition and so an agent SHALL always
4320 return a value of '-1' to indicate 'other' for the value of the
4321 jmAttributeValueAsInteger object for these attributes.

4322

4323 For attributes which do have the 'INTEGER:' tag in the
4324 JmAttributeTypeTC definition, if the integer value is not (yet)
4325 known, the agent either (1) SHALL not materialize the row in
4326 the jmAttributeTable until the value is known or (2) SHALL
4327 return a '-2' to represent an 'unknown' counting integer value,
4328 a '0' to represent an 'unknown' index value, and a '2' to
4329 represent an 'unknown(2)' enum value."

```
4330     DEFVAL      { -2 }      -- default value is unknown(-2)
4331     ::= { jmAttributeEntry 3 }
```

4332

```
4333 jmAttributeValueAsOctets OBJECT-TYPE
4334     SYNTAX      OCTET STRING(SIZE(0..63))
4335     MAX-ACCESS  read-only
4336     STATUS      current
4337     DESCRIPTION
4338         "The octet string value of the attribute.  The value of the
4339         attribute SHALL be represented as an OCTET STRING if the enum
4340         description in the JmAttributeTypeTC textual-convention
4341         definition has the tag: 'OCTETS:'."
4342
4343         Depending on the enum definition, this object value MAY be a
4344         coded character set string (text), such as 'JmUTF8StringTC', or
4345         a binary octet string, such as 'DateAndTime'.
4346
4347         Attributes for which the concept of an octet string value is
4348         meaningless, such as pagesCompleted, do not have the tag
4349         'OCTETS:' in the JmAttributeTypeTC definition and so the agent
4350         SHALL always return a zero length string for the value of the
4351         jmAttributeValueAsOctets object.
4352
4353         For attributes which do have the 'OCTETS:' tag in the
4354         JmAttributeTypeTC definition, if the OCTET STRING value is not
4355         (yet) known, the agent either SHALL NOTnot materialize the row
4356         in the jmAttributeTable until the value is known or SHALL
4357         return a zero-length string."
4358     DEFVAL      { 'H' } -- empty string
4359     ::= { jmAttributeEntry 4 }
```

```
4361 -- Notifications and Trapping
4362 -- Reserved for the future
4363
4364 jobmonMIBNotifications OBJECT IDENTIFIER ::= { jobmonMIB 2 }
4365
4366
4367
4368 -- Conformance Information
4369
4370 jmMIBConformance OBJECT IDENTIFIER ::= { jobmonMIB 3 }
4371
4372
4373
4374 -- compliance statements
4375 jmMIBCompliance MODULE-COMPLIANCE
4376     STATUS current
4377     DESCRIPTION
4378         "The compliance statement for agents that implement the
4379         job monitoring MIB."
4380     MODULE -- this module
4381     MANDATORY-GROUPS {
4382         jmGeneralGroup, jmJobIDGroup, jmJobGroup, jmAttributeGroup }
4383
4384     OBJECT jmGeneralJobSetName
4385     SYNTAX JmUTF8StringTC (SIZE(0..8))
4386     DESCRIPTION
4387         "Only 8 octets maximum string length NEED be supported by the
4388         agent."
4389
4390     OBJECT jmJobOwner
4391     SYNTAX JmJobStringTC (SIZE(0..16))
4392     DESCRIPTION
4393         "Only 16 octets maximum string length NEED be supported by the
4394         agent."
4395
4396 -- There are no CONDITIONALLY MANDATORY or OPTIONAL groups.
4397
4398 ::= { jmMIBConformance 1 }
4399
```



```
4400 jmMIBGroups      OBJECT IDENTIFIER ::= { jmMIBConformance 2 }
4401
4402 jmGeneralGroup OBJECT-GROUP
4403     OBJECTS {
4404         jmGeneralNumberOfActiveJobs,    jmGeneralOldestActiveJobIndex,
4405         jmGeneralNewestActiveJobIndex,  jmGeneralJobPersistence,
4406         jmGeneralAttributePersistence,  jmGeneralJobSetName}
4407     STATUS current
4408     DESCRIPTION
4409         "The general group."
4410     ::= { jmMIBGroups 1 }
4411
4412
4413
4414 jmJobIDGroup OBJECT-GROUP
4415     OBJECTS {
4416         jmJobIDJobSetIndex, jmJobIDJobIndex }
4417     STATUS current
4418     DESCRIPTION
4419         "The job ID group."
4420     ::= { jmMIBGroups 2 }
4421
4422
4423
4424 jmJobGroup OBJECT-GROUP
4425     OBJECTS {
4426         jmJobState, jmJobStateReasons1, jmNumberOfInterveningJobs,
4427         jmJobKOctetsPerCopyRequested, jmJobKOctetsProcessed,
4428         jmJobImpressionsPerCopyRequested, jmJobImpressionsCompleted,
4429         jmJobOwner }
4430     STATUS current
4431     DESCRIPTION
4432         "The job group."
4433     ::= { jmMIBGroups 3 }
4434
4435
4436
4437 jmAttributeGroup OBJECT-GROUP
4438     OBJECTS {
4439         jmAttributeValueAsInteger, jmAttributeValueAsOctets }
4440     STATUS current
4441     DESCRIPTION
4442         "The attribute group."
4443     ::= { jmMIBGroups 4 }
4444
4445
4446 END
```

4447 5 Appendix A - Implementing the Job Life Cycle

4448 The job object has well-defined states and client operations that
4449 affect the transition between the job states. Internal server and
4450 device actions also affect the transitions of the job between the job
4451 states. These states and transitions are referred to as the job's *life*
4452 *cycle*.

4453 Not all implementations of job submission protocols have all of the
4454 states of the job model specified here. The job model specified here
4455 is intended to be a superset of most implementations. It is the
4456 purpose of the agent to map the particular implementation's job life
4457 cycle onto the one specified here. The agent MAY omit any states not
4458 implemented. Only the processing and completed states are required to
4459 be implemented by an agent. However, a conforming management
4460 application SHALL be prepared to accept any of the states in the job
4461 life cycle specified here, so that the management application can
4462 interoperate with any conforming agent.

4463 The job states are intended to be user visible. The agent SHALL make
4464 these states visible in the MIB, but only for the subset of job states
4465 that the implementation has. Some implementations MAY need to have
4466 sub-states of these user-visible states. The jmJobStateReasons1 object
4467 and the jobStateReasonsN ($N=2..4$) attributes can be used to represent
4468 the sub-states of the jobs.

4469 Job states are intended to last a user-visible length of time in most
4470 implementations. However, some jobs may pass through some states in
4471 zero time in some situations and/or in some implementations.

4472 The job model does not specify how accounting and auditing is
4473 implemented, except to assume that accounting and auditing logs are
4474 separate from the job life cycle and last longer than job entries in
4475 the MIB. Jobs in the completed, aborted, or canceled states are not
4476 logs, since jobs in these states are accessible via SNMP protocol
4477 operations and SHALL be removed from the Job Monitoring MIB tables
4478 after a site-settable or implementation-defined period of time. An
4479 accounting application MAY copy accounting information incrementally to
4480 an accounting log as a job processes, or MAY be copied while the job is
4481 in the canceled, aborted, or completed states, depending on
4482 implementation. The same is true for auditing logs.

4483 The jmJobState object specifies the standard job states. The normal
4484 job state transitions are shown in the state transition diagram
4485 presented in Table 1.

4486 6 APPENDIX B - Support of Job Submission Protocols

4487 A companion PWG document, entitled "Job Submission Protocol Mapping
4488 Recommendations for the Job Monitoring MIB" [protomap] contains the
4489 recommended usage of each of the objects and attributes in this MIB
4490 with a number of job submission protocols. In particular, which job
4491 submission ID format should be used is indicated for each job
4492 submission protocol.

4493 Some job submission protocols have support for the client to specify a
4494 job submission ID. A second approach is to enhance the document format
4495 to embed the job submission ID in the document data. This second
4496 approach is independent of the job submission protocol. This appendix
4497 lists some examples of these approaches.

4498 Some PJL implementations wrap a banner page as a PJL job around a job
4499 submitted by a client. If this results in multiple job submission IDs,
4500 the agent SHALL create multiple jmJobIDEntry rows in the jmJobIDTable
4501 that each point to the same job entry in the job tables. See the
4502 specification of the jmJobIDEntry.

4503 7 References

4504 [char-set-policy] Harald Avelstrand, "IETF Policy on Character Sets and
4505 Language", June 1997. Latest draft: <draft-avelstrand-charset-
4506 policy-00.txt>

4507 [GB2312] GB 2312-1980, "Chinese People's Republic of China (PRC) mixed
4508 one byte and two byte coded character set"

4509 [hr-mib] P. Grillo, S. Waldbusser, "Host Resources MIB", RFC 1514,
4510 September 1993

4511 [iana] J. Reynolds, and J. Postel, "Assigned Numbers", STD 2, RFC 1700,
4512 ISI, October 1994.

4513 [IANA-charsets] Coded Character Sets registered by IANA and assigned an
4514 enum value for use in the CodedCharSet textual convention imported from
4515 the Printer MIB. See ftp://ftp.isi.edu/in-
4516 notes/iana/assignments/character-sets

4517 [iana-media-types] IANA Registration of MIME media types (MIME content
4518 types/subtypes). See ftp://ftp.isi.edu/in-notes/iana/assignments/

4519 [ipp-model] Internet Printing Protocol/1.0: Model and Semantics, work
4520 in progress on the IETF standards track. See draft-ietf-ipp-model-
4521 09.txt. See also http://www.pwg.org/ipp/index.html

4522 [ISO-639] ISO 639:1988 (E/F) - Code for Representation of names of
4523 languages - The International Organization for Standardization, 1st
4524 edition, 1988.

- 4525 [ISO 646] ISO/IEC 646:1991, "Information technology -- ISO 7-bit coded
4526 character set for information interchange", JTC1/SC2.
- 4527 [ISO 8859] ISO/IEC 8859-1:1987, "Information technology -- 8-bit single
4528 byte coded graphic character sets - Part 1: Latin alphabet No. 1,
4529 JTC1/SC2."
- 4530 [ISO 2022] ISO/IEC 2022:1994 - "Information technology -- Character
4531 code structure and extension techniques", JTC1/SC2.
- 4532 [ISO-3166] ISO 3166:1988 (E/F) - Codes for representation of names of
4533 countries - The International Organization for Standardization, 3rd
4534 edition, 1988-08-15."
- 4535 [ISO-10646] ISO/IEC 10646-1:1993, "Information technology -- Universal
4536 Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and
4537 Basic Multilingual Plane, JTC1/SC2.
- 4538 [iso-dpa] ISO/IEC 10175 Document Printing Application (DPA). See
4539 <ftp://ftp.pwg.org/pub/pwg/dpa/>
- 4540 [JIS X0208] JIS X0208-1990, "Japanese two byte coded character set."
- 4541 [mib-II] MIB-II, RFC 1213.
- 4542 [print-mib] Smith, R., Wright, F., Hastings, T., Zilles, S. and
4543 Gyllenskog, J., "Printer MIB", RFC 1759, proposed IETF standard, March
4544 1995. See also [print-mib-draft].
- 4545 [print-mib-draft] Turner, R., "Printer MIB", work in progress, on the
4546 standards track as a draft standard: <draft-ietf-printmib-mib-info-
4547 02.txt>, October 15, 1997.
- 4548 [protomap] Bergman, R., "Job Submission Protocol Mapping
4549 Recommendations for the Job Monitoring MIB," work in progress as an
4550 informational RFC. See <draft-bergman-printmib-job-protomap-01.txt>,
4551 January 12, 1998.
- 4552 [pwg] The Printer Working Group is a printer industry consortium open
4553 to any individuals. For more information, access the PWG web page:
4554 <http://www.pwg.org>
- 4555 [req-words] S. Bradner, "Keywords for use in RFCs to Indicate
4556 Requirement Levels", RFC 2119, March 1997.
- 4557 [rfc 1738] Berners-Lee, T., Masinter, L., McCahill, M., "Uniform
4558 Resource Locators (URL)", RFC 1738, December 1994.
- 4559 [RFC-1766] Avelstrand, H., "Tags for the Identification of Languages",
4560 RFC 1766, March 1995.

4561 [rfc 2130] C. Weider, C. Preston, K. Simonsen, H. Alvestrand, R.
4562 Atkinson, M. Crispin, and P. Svanberg, "The Report of the IAB Character
4563 Set Workshop held 29 Feb-1 March, 1997", April 1997, RFC 2130.

4564 [SMIv2-SMI] J. Case, et al. "Structure of Management Information for
4565 Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC
4566 1902, January 1996.

4567 [SMIv2-TC] J. Case, et al. "Textual Conventions for Version 2 of the
4568 Simple Network Management Protocol (SNMPv2)", RFC 1903, January 1996.

4569 [tipsi] IEEE 1284.1, Transport-independent Printer System Interface
4570 (TIPSI).

4571 [URI-spec] Berners-Lee, T., Masinter, L., McCahill, M. , "Uniform
4572 Resource Locators (URL)", RFC 1738, December, 1994.

4573 [US-ASCII] Coded Character Set - 7-bit American Standard Code for
4574 Information Interchange, ANSI X3.4-1986.

4575 [UTF-8] F. Yergeau, "UTF-8, a transformation format of Unicode and ISO
4576 10646", RFC 2044, October 1996.

4577 8 Author's Addresses
4578 Ron Bergman
4579 Dataproducts Corp.
4580 1757 Tapo Canyon Road
4581 Simi Valley, CA 93063-3394
4582
4583 Phone: 805-578-4421
4584 Fax: 805-578-4001
4585 Email: rbergman@dpc.com
4586
4587
4588 Tom Hastings
4589 Xerox Corporation, ESAE-231
4590 701 S. Aviation Blvd.
4591 El Segundo, CA 90245
4592
4593 Phone: 310-333-6413
4594 Fax: 310-333-5514
4595 EMail: hasting@cpl0.es.xerox.com
4596
4597
4598 Scott A. Isaacson
4599 Novell, Inc.
4600 122 E 1700 S
4601 Provo, UT 84606
4602
4603 Phone: 801-861-7366
4604 Fax: 801-861-4025

4605 EMail: scott_isaacson@novell.com

4606

4607

4608 Harry Lewis

4609 IBM Corporation

4610 6300 Diagonal Hwy

4611 Boulder, CO 80301

4612

4613 Phone: (303) 924-5337

4614 Fax:

4615 Email: harryl@us.ibm.com

4616

4617

4618 Send questions and comments to the Printer Working Group (PWG)
4619 using the Job Monitoring Project (JMP) Mailing List: jmp@pwg.org

4620

4621 To learn how to subscribe, send email to: jmp-request@pwg.org

4622

4623 Implementers of this specification are encouraged to join the jmp
4624 mailing list in order to participate in discussions on any
4625 clarifications needed and registration proposals for additional
4626 attributes and values being reviewed in order to achieve consensus.

4627

4628 For further information, access the PWG web page under "JMP":

4629

4630 <http://www.pwg.org/>

4631

4632 Other Participants:

4633 Chuck Adams - Tektronix

4634 Jeff Barnett - IBM

4635 Keith Carter, IBM Corporation

4636 Jeff Copeland - QMS

4637 Andy Davidson - Tektronix

4638 Roger deBry - IBM

4639 Mabry Dozier - QMS

4640 Lee Faerrell - Canon

4641 Steve Gebert - IBM

4642 Robert Herriot - Sun Microsystems Inc.

4643 Shige Kanemitsu - Kyocera

4644 David Kellerman - Northlake Software

4645 Rick Landau - Digital

4646 Pete Loya - HP

4647 Ray Lutz - Cognisys

4648 Jay Martin - Underscore

4649 Mike MacKay, Novell, Inc.

4650 Stan McConnell - Xerox

4651 Carl-Uno Manros, Xerox, Corp.

4652 Pat Nogay - IBM

4653 Bob Pentecost - HP

4654 Rob Rhoads - Intel

4655 David Roach - Unisys
4656 Stuart Rowley - Kyocera
4657 Hiroyuki Sato - Canon
4658 Bob Setterbo - Adobe
4659 Gail Songer, EFI
4660 Mike Timperman - Lexmark
4661 Randy Turner - Sharp
4662 William Wagner - Digital Products
4663 Jim Walker - Dazel
4664 Chris Wellens - Interworking Labs
4665 Rob Whittle - Novell
4666 Don Wright - Lexmark
4667 Lloyd Young - Lexmark
4668 Atsushi Yuki - Kyocera
4669 Peter Zehler, Xerox, Corp.

4670 9 Change History

4671 This section summarizes the changes in each version after version 1.0
4672 in reverse chronological order.

4673 9.1 Changes to produce version 1.1, dated October 1, 1998

4674 The following changes were made to version 1.0, dated February 3, 1998
4675 to make version 1.1, dated October 1, 1998:

4676 1. Clarified sections 3.3.3 and 3.3.7 so that the DEFVAL of 0 for index
4677 attributes is different from the DEFVAL for
4678 jmAttributeValueAsInteger which is -2.

4679 2. Clarified the relationships of the values of the
4680 JmJobCollationTypeTC with the IPP "multiple-document-handling"
4681 attribute.

4682 3. Clarified that the values of the mediumRequested(170) and
4683 mediumConsumed(171) attributes may be any of the IPP 'media' values
4684 which are media names, media size names, and input tray names.

4685 4. Added the two attributes approved by the PWG for registration in
4686 April 1998: mediumTypeConsumed(174) and mediumSizeConsumed(175).

4687 5. Changed "insure" to "ensure'.

4688 6. Correct an incorrect reference in the jmAttributeEntry DESCRIPTION
4689 from jmJobTable to jmAttributeTable.

- 4690 9.2 Changes to produce version 1.2, dated October 2, 1998
- 4691 The following changes were made to version 1.1, dated October 1, 1998
4692 to make version 1.2, dated October 2, 1998:
- 4693 1. Removed all REFERENCE clauses since they referred to sections in the
4694 specification that were not in the MIB.
- 4695 2. Moved the definitions of the attributes from the TC to a new section
4696 3.3.8.
- 4697 3. Removed the attributes from the Table of Contents
- 4698 4. Added the data types as ASN.1 comments after each attribute enum.
- 4699 5. Changed a number of occurrences of "SHALL" to "is" when they were
4700 just definitions, rather than conformance requirements.
- 4701

4702 10 INDEX

4703 This index includes the textual conventions, the objects, and the
4704 attributes. Textual conventions all start with the prefix: "JM" and
4705 end with the suffix: "TC". Objects all starts with the prefix: "jm"
4706 followed by the group name. Attributes are identified with enums, and
4707 so start with any lower case letter and have no special prefix.

4708
4709 colorantConsumed, 40
4710 colorantRequested, 40
4711 deviceNameRequested, 30
4712 documentCopiesCompleted, 35
4713 documentCopiesRequested, 35
4714 documentFormat, 32
4715 documentFormatIndex, 31
4716 documentName, 31
4717 fileName, 31
4718 finishing, 34
4719 fullColorImpressionsCompleted, 37
4720 highlightColorImpressionsCompleted, 38
4721 impressionsCompletedCurrentCopy, 37
4722 impressionsInterpreted, 37
4723 impressionsSentToDevice, 37
4724 impressionsSpooled, 37
4725 jmAttributeInstanceIndex, 106
4726 jmAttributeTypeIndex, 106
4727 JmAttributeTypeTC, 72
4728 jmAttributeValueAsInteger, 107
4729 jmAttributeValueAsOctets, 108
4730 JmBooleanTC, 61
4731 JmFinishingTC, 59
4732 jmGeneralAttributePersistence, 92
4733 jmGeneralJobPersistence, 92
4734 jmGeneralJobSetIndex, 90
4735 jmGeneralJobSetName, 93
4736 jmGeneralNewestActiveJobIndex, 91
4737 jmGeneralNumberOfActiveJobs, 90
4738 jmGeneralOldestActiveJobIndex, 91
4739 JmJobCollationTypeTC, 64
4740 jmJobIDJobIndex, 96
4741 jmJobIDJobSetIndex, 96
4742 jmJobImpressionsCompleted, 102
4743 jmJobImpressionsPerCopyRequested, 102
4744 jmJobIndex, 99
4745 jmJobKOctetsPerCopyRequested, 101
4746 jmJobKOctetsProcessed, 101
4747 jmJobOwner, 103
4748 JmJobServiceTypesTC, 77
4749 JmJobSourcePlatformTypeTC, 57
4750 jmJobState, 99
4751 jmJobStateReasons1, 100

4752 JmJobStateReasons1TC, 78
4753 JmJobStateReasons2TC, 82
4754 JmJobStateReasons3TC, 87
4755 JmJobStateReasons4TC, 87
4756 JmJobStateTC, 69
4757 JmJobStringTC, 56
4758 jmJobSubmissionID, 95
4759 JmJobSubmissionIDTypeTC, 64
4760 JmMediumTypeTC, 61
4761 JmNaturalLanguageTagTC, 56
4762 jmNumberOfInterveningJobs, 100
4763 JmPrinterResolutionTC, 60
4764 JmPrintQualityTC, 60
4765 JmTimeStampTC, 57
4766 JmTonerEconomyTC, 61
4767 JmUTF8StringTC, 56
4768 jobAccountName, 27
4769 jobCodedCharSet, 26
4770 jobCollationType, 36
4771 jobComment, 31
4772 jobCompletionTime, 43
4773 jobCopiesCompleted, 35
4774 jobCopiesRequested, 35
4775 jobHold, 33
4776 jobHoldUntil, 33
4777 jobKOctetsTransferred, 36
4778 jobName, 28
4779 jobNaturalLanguageTag, 26
4780 jobOriginatingHost, 30
4781 jobPriority, 32
4782 jobProcessAfterDateAndTime, 33
4783 jobProcessingCPUTime, 43
4784 jobServiceTypes, 29
4785 jobSourceChannelIndex, 29
4786 jobSourcePlatformType, 29
4787 jobStartedBeingHeldTime, 43
4788 jobStartedProcessingTime, 43
4789 jobStateReasons2, 25
4790 jobStateReasons3, 25
4791 jobStateReasons4, 25
4792 jobSubmissionTime, 42
4793 jobSubmissionToServerTime, 42
4794 jobURI, 27
4795 mediumConsumed, 40
4796 mediumRequested, 40
4797 mediumSizeConsumed, 41
4798 mediumTypeConsumed, 41
4799 numberOfDocuments, 30
4800 other, 24
4801 outputBin, 33
4802 pagesCompleted, 38
4803 pagesCompletedCurrentCopy, 39

4804 pagesRequested, 38
4805 physicalDevice, 30
4806 printerResolutionRequested, 34
4807 printerResolutionUsed, 34
4808 printQualityRequested, 34
4809 printQualityUsed, 34
4810 processingMessage, 25
4811 processingMessageNaturalLangTag, 26
4812 queueNameRequested, 30
4813 serverAssignedJobName, 28
4814 sheetCompletedCopyNumber, 36
4815 sheetCompletedDocumentNumber, 36
4816 sheetsCompleted, 39
4817 sheetsCompletedCurrentCopy, 39
4818 sheetsRequested, 39
4819 sides, 34
4820 submittingApplicationName, 29
4821 submittingServerName, 29
4822 tonerDensityRequested, 34
4823 tonerDensityUsed, 35
4824 tonerEcomonyRequested, 34
4825 tonerEcomonyUsed, 34

4826