

1 INTERNET-DRAFT

R. Bergman
Dataproducts Corp.
T. Hastings
Xerox Corporation
S. Isaacson
Novell, Inc.
H. Lewis
IBM Corp.
January 13, 1998

10 Job Monitoring MIB - V1
11 <draft-ietf-printmib-job-monitor-07.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 July 13, 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 | 8 |
| 48 | 1.1 Types of Information in the MIB | 8 |
| 49 | 1.2 Types of Job Monitoring Applications | 10 |
| 50 | 2. TERMINOLOGY AND JOB MODEL | 11 |
| 51 | 2.1 System Configurations for the Job Monitoring MIB | 14 |
| 52 | 2.1.1 Configuration 1 - client-printer | 14 |
| 53 | 2.1.2 Configuration 2 - client-server-printer - agent in the | |
| 54 | server | 15 |
| 55 | 2.1.3 Configuration 3 - client-server-printer - client monitors | |
| 56 | printer agent and server | 16 |
| 57 | 3. MANAGED OBJECT USAGE | 18 |
| 58 | 3.1 Conformance Considerations | 18 |
| 59 | 3.1.1 Conformance Terminology | 18 |
| 60 | 3.1.2 Agent Conformance Requirements | 18 |
| 61 | 3.1.2.1 MIB II System Group objects | 19 |
| 62 | 3.1.2.2 MIB II Interface Group objects | 19 |
| 63 | 3.1.2.3 Printer MIB objects | 19 |
| 64 | 3.1.3 Job Monitoring Application Conformance Requirements | 19 |
| 65 | 3.2 The Job Tables and the Oldest Active and Newest Active Indexes | 19 |
| 66 | 3.3 The Attribute Mechanism | 21 |
| 67 | 3.3.1 Conformance of Attribute Implementation | 22 |
| 68 | 3.3.2 Useful, 'Unknown', and 'Other' Values for Objects and | |
| 69 | Attributes | 22 |
| 70 | 3.3.3 Data Sub-types and Attribute Naming Conventions | 23 |
| 71 | 3.3.4 Single-Value (Row) Versus Multi-Value (MULTI-ROW) | |
| 72 | Attributes | 24 |
| 73 | 3.3.5 Requested Objects and Attributes | 24 |
| 74 | 3.3.6 Consumption Attributes | 25 |
| 75 | 3.3.7 Index Value Attributes | 25 |
| 76 | 3.4 Monitoring Job Progress | 25 |
| 77 | 3.5 Job Identification | 29 |
| 78 | 3.6 Internationalization Considerations | 30 |
| 79 | 3.6.1 Text generated by the server or device | 30 |
| 80 | 3.6.2 Text supplied by the job submitter | 31 |
| 81 | 3.6.3 'DateAndTime' for representing the date and time | 32 |

| | | |
|-----|--|----|
| 82 | 3.7 IANA and PWG Registration Considerations | 32 |
| 83 | 3.7.1 PWG Registration of enums | 33 |
| 84 | 3.7.1.1 Type 1 enumerations | 33 |
| 85 | 3.7.1.2 Type 2 enumerations | 33 |
| 86 | 3.7.1.3 Type 3 enumeration | 34 |
| 87 | 3.7.2 PWG Registration of type 2 bit values | 34 |
| 88 | 3.7.3 PWG Registration of Job Submission Id Formats | 34 |
| 89 | 3.7.4 PWG Registration of MIME types/sub-types for document- | |
| 90 | formats 34 | |
| 91 | 3.8 Security Considerations | 34 |
| 92 | 3.8.1 Read-Write objects | 34 |
| 93 | 3.8.2 Read-Only Objects In Other User's Jobs | 35 |
| 94 | 3.9 Notifications | 35 |
| 95 | 4. MIB SPECIFICATION | 35 |
| 96 | Textual Conventions for this MIB Module | 37 |
| 97 | JmUTF8StringTC | 37 |
| 98 | JmJobStringTC | 37 |
| 99 | JmNaturalLanguageTagTC | 37 |
| 100 | JmTimeStampTC | 37 |
| 101 | JmJobSourcePlatformTypeTC | 38 |
| 102 | JmFinishingTC | 39 |
| 103 | JmPrintQualityTC | 40 |
| 104 | JmPrinterResolutionTC | 40 |
| 105 | JmTonerEconomyTC | 41 |
| 106 | JmBooleanTC | 41 |
| 107 | JmMediumTypeTC | 41 |
| 108 | JmJobCollationTypeTC | 43 |
| 109 | JmJobStateTC | 47 |
| 110 | JmAttributeTypeTC | 50 |
| 111 | other (Int32(-2..) and/or Octets63) | 50 |
| 112 | Job State attributes | 51 |
| 113 | jobStateReasons2 (JmJobStateReasons2TC) | 51 |
| 114 | jobStateReasons3 (JmJobStateReasons3TC) | 51 |
| 115 | jobStateReasons4 (JmJobStateReasons4TC) | 51 |
| 116 | processingMessage (UTF8String63) | 51 |
| 117 | processingMessageNaturalLangTag (Octets63) | 52 |
| 118 | jobCodedCharSet (CodedCharSet) | 52 |
| 119 | jobNaturalLanguageTag (Octets63) | 53 |
| 120 | Job Identification attributes | 53 |
| 121 | jobURI (Octets(0..63)) | 53 |
| 122 | jobAccountName (Octets63) | 53 |
| 123 | serverAssignedJobName (JobString63) | 54 |
| 124 | jobName (JobString63) | 54 |
| 125 | jobServiceTypes (JmJobServiceTypesTC) | 55 |
| 126 | jobSourceChannelIndex (Int32(0..)) | 55 |
| 127 | jobSourcePlatformType (JmJobSourcePlatformTypeTC) | 55 |
| 128 | submittingServerName (JobString63) | 55 |

| | | |
|-----|---|----|
| 129 | submittingApplicationName (JobString63) | 55 |
| 130 | jobOriginatingHost (JobString63) | 56 |
| 131 | deviceNameRequested (JobString63) | 56 |
| 132 | queueNameRequested (JobString63) | 56 |
| 133 | physicalDevice (hrDeviceIndex and/or UTF8String63) | 56 |
| 134 | numberOfDocuments (Int32(-2..)) | 56 |
| 135 | fileName (JobString63) | 57 |
| 136 | documentName (JobString63) | 57 |
| 137 | jobComment (JobString63) | 57 |
| 138 | documentFormatIndex (Int32(0..)) | 57 |
| 139 | documentFormat (PrtInterpreterLangFamilyTC and/or Octets63) | 58 |
| 140 | Job Parameter attributes | 58 |
| 141 | jobPriority (Int32(-2..100)) | 58 |
| 142 | jobProcessAfterDateAndTime (DateAndTime) | 59 |
| 143 | jobHold (JmBooleanTC) | 59 |
| 144 | jobHoldUntil (JobString63) | 59 |
| 145 | outputBin (Int32(0..) and/or JobString63) | 59 |
| 146 | sides (Int32(-2..2)) | 60 |
| 147 | finishing (JmFinishingTC) | 60 |
| 148 | Image Quality attributes (requested and used) | 60 |
| 149 | printQualityRequested (JmPrintQualityTC) | 60 |
| 150 | printQualityUsed (JmPrintQualityTC) | 60 |
| 151 | printerResolutionRequested (JmPrinterResolutionTC) | 60 |
| 152 | printerResolutionUsed (JmPrinterResolutionTC) | 60 |
| 153 | tonerEcomonyRequested (JmTonerEconomyTC) | 60 |
| 154 | tonerEcomonyUsed (JmTonerEconomyTC) | 60 |
| 155 | tonerDensityRequested (Int32(-2..100)) | 60 |
| 156 | tonerDensityUsed (Int32(-2..100)) | 61 |
| 157 | Job Progress attributes (requested and consumed) | 61 |
| 158 | jobCopiesRequested (Int32(-2..)) | 61 |
| 159 | jobCopiesCompleted (Int32(-2..)) | 61 |
| 160 | documentCopiesRequested (Int32(-2..)) | 61 |
| 161 | documentCopiesCompleted (Int32(-2..)) | 61 |
| 162 | jobKOctetsTransferred (Int32(-2..)) | 62 |
| 163 | sheetCompletedCopyNumber (Int32(-2..))4 | 62 |
| 164 | sheetCompletedDocumentNumber (Int32(-2..))4 | 62 |
| 165 | jobCollationType JmJobCollationTypeTC) | 62 |
| 166 | Impression attributes (requested and consumed) | 63 |
| 167 | impressionsSpooled (Int32(-2..)) | 63 |
| 168 | impressionsSentToDevice (Int32(-2..)) | 63 |
| 169 | impressionsInterpreted (Int32(-2..)) | 63 |
| 170 | impressionsCompletedCurrentCopy (Int32(-2..)) | 63 |
| 171 | fullColorImpressionsCompleted (Int32(-2..)) | 63 |
| 172 | highlightColorImpressionsCompleted (Int32(-2..)) | 64 |
| 173 | Page attributes (requested and consumed) | 64 |
| 174 | pagesRequested (Int32(-2..)) | 64 |
| 175 | pagesCompleted (Int32(-2..)) | 64 |
| 176 | pagesCompletedCurrentCopy (Int32(-2..)) | 65 |
| 177 | Sheet attributes (requested and consumed) | 65 |
| 178 | sheetsRequested (Int32(-2..)) | 65 |
| 179 | sheetsCompleted (Int32(-2..)) | 65 |
| 180 | sheetsCompletedCurrentCopy (Int32(-2..)) | 65 |

| | | |
|-----|--|-----|
| 181 | Resource attributes (requested and consumed) | 65 |
| 182 | mediumRequested (JmMediumTypeTC and/or JobString63) | 66 |
| 183 | mediumConsumed (Int32(-2..) and/or JobString63) | 66 |
| 184 | colorantRequested (Int32(-2..) and/or JobString63) | 66 |
| 185 | colorantConsumed (Int32(-2..) and/or JobString63) | 67 |
| 186 | Time attributes (set by server or device) | 67 |
| 187 | jobSubmissionToServerTime (JmTimeStampTC and/or DateAndTime) | 67 |
| 188 | jobSubmissionTime (JmTimeStampTC and/or DateAndTime) | 67 |
| 189 | jobStartedBeingHeldTime (JmTimeStampTC) | 68 |
| 190 | jobStartedProcessingTime (JmTimeStampTC and/or DateAndTime) | 68 |
| 191 | jobCompletionTime (JmTimeStampTC and/or DateAndTime) | 68 |
| 192 | jobProcessingCPUTime (Int32(-2..)) | 68 |
| 193 | JmJobServiceTypesTC | 70 |
| 194 | JmJobStateReasons1TC | 72 |
| 195 | JmJobStateReasons2TC | 76 |
| 196 | JmJobStateReasons3TC | 80 |
| 197 | JmJobStateReasons4TC | 80 |
| 198 | The General Group (MANDATORY) | 81 |
| 199 | jmGeneralJobSetIndex (Int32(1..32767)) | 82 |
| 200 | jmGeneralNumberOfActiveJobs (Int32(0..)) | 82 |
| 201 | jmGeneralOldestActiveJobIndex (Int32(0..)) | 83 |
| 202 | jmGeneralNewestActiveJobIndex (Int32(0..)) | 83 |
| 203 | jmGeneralJobPersistence (Int32(15..)) | 84 |
| 204 | jmGeneralAttributePersistence (Int32(15..)) | 84 |
| 205 | jmGeneralJobSetName (UTF8String63) | 85 |
| 206 | The Job ID Group (MANDATORY) | 85 |
| 207 | jmJobSubmissionID (OCTET STRING(SIZE(48))) | 87 |
| 208 | jmJobIDJobSetIndex (Int32(0..32767)) | 88 |
| 209 | jmJobIDJobIndex (Int32(0..)) | 88 |
| 210 | The Job Group (MANDATORY) | 88 |
| 211 | jmJobIndex (Int32(1..)) | 90 |
| 212 | jmJobState (JmJobStateTC) | 90 |
| 213 | jmJobStateReasons1 (JmJobStateReasons1TC) | 91 |
| 214 | jmNumberOfInterveningJobs (Int32(-2..)) | 91 |
| 215 | jmJobKOctetsPerCopyRequested (Int32(-2..)) | 92 |
| 216 | jmJobKOctetsProcessed (Int32(-2..)) | 92 |
| 217 | jmJobImpressionsPerCopyRequested (Int32(-2..)) | 93 |
| 218 | jmJobImpressionsCompleted (Int32(-2..)) | 93 |
| 219 | jmJobOwner (JobString63) | 94 |
| 220 | The Attribute Group (MANDATORY) | 94 |
| 221 | jmAttributeTypeIndex (JmAttributeTypeTC) | 97 |
| 222 | jmAttributeInstanceIndex (Int32(1..32767)) | 97 |
| 223 | jmAttributeValueAsInteger (Int32(-2..)) | 98 |
| 224 | jmAttributeValueAsOctets (Octets63) | 99 |
| 225 | 5. APPENDIX A - IMPLEMENTING THE JOB LIFE CYCLE | 102 |
| 226 | 6. APPENDIX B - SUPPORT OF JOB SUBMISSION PROTOCOLS | 103 |
| 227 | 7. REFERENCES | 103 |

| | | | |
|-----|----|--------------------|-----|
| 228 | 8. | AUTHOR'S ADDRESSES | 105 |
| 229 | 9. | INDEX | 108 |
| 230 | | | |

231 Job Monitoring MIB

232 1. Introduction

233 This specification defines an official Printer Working Group (PWG)
234 [PWG] standard SNMP MIB for the monitoring of jobs on network printers.
235 This specification is being published as an IETF Information Document
236 for the convenience of the Internet community. In consultation with
237 the IETF Application Area Directors, it was concluded properly belongs
238 as an Information document, because this MIB monitors a service node on
239 the network, rather than a network node proper.

240 The Job Monitoring MIB is intended to be implemented by an agent within
241 a printer or the first server closest to the printer, where the printer
242 is either directly connected to the server only or the printer does not
243 contain the job monitoring MIB agent. It is recommended that
244 implementations place the SNMP agent as close as possible to the
245 processing of the print job. This MIB applies to printers with and
246 without spooling capabilities. This MIB is designed to be compatible
247 with most current commonly-used job submission protocols. In most
248 environments that support high function job submission/job control
249 protocols, like ISO DPA[iso-dpa], those protocols would be used to
250 monitor and manage print jobs rather than using the Job Monitoring MIB.

251 The Job Monitoring MIB consists of a General Group, a Job Submission ID
252 Group, a Job Group, and an Attribute Group. Each group is a table.
253 All accessible objects are read-only. The General Group contains
254 general information that applies to all jobs in a job set. The Job
255 Submission ID table maps the job submission ID that the client uses to
256 identify a job to the jmJobIndex that the Job Monitoring Agent uses to
257 identify jobs in the Job and Attribute tables. The Job table contains
258 the MANDATORY integer job state and status objects. The Attribute
259 table consists of multiple entries per job that specify (1) job and
260 document identification and parameters, (2) requested resources, and
261 (3) consumed resources during and after job processing/printing. A
262 larger number of job attributes are defined as textual conventions that
263 an agent SHALL return if the server or device implements the
264 functionality so represented and the agent has access to the
265 information.

266 1.1 Types of Information in the MIB

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

270 User:

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

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

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

279 Provide error and diagnostic information for jobs that did not
280 successfully complete.

281 Operator:

282 Provide a presentation of the state of all the jobs in the print
283 system.

284 Provide the ability to identify the user that submitted the print
285 job.

286 Provide the ability to identify the resources required by each
287 job.

288 Provide the ability to define which physical printers are
289 candidates for the print job.

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

294 Capacity Planner:

295 Provide the ability to determine printer utilization as a
296 function of time.

297 Provide the ability to determine how long jobs wait before
298 starting to print.

299 Accountant:

300 Provide information to allow the creation of a record of
301 resources consumed and printer usage data for charging users or
302 groups for resources consumed.

303 Provide information to allow the prediction of consumable usage
304 and resource need.

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

311 1.2 Types of Job Monitoring Applications

312 The Job Monitoring MIB is designed for the following types of
313 monitoring applications:

- 314 1. Monitor a single job starting when the job is submitted and
315 ending a defined period after the job completes. The Job
316 Submission ID table provides the map to find the specific job
317 to be monitored.
- 318 2. Monitor all 'active' jobs in a queue, which this specification
319 generalizes to a "job set". End users may use such a program
320 when selecting a least busy printer, so the MIB is designed for
321 such a program to start up quickly and find the information
322 needed quickly without having to read all (completed) jobs in
323 order to find the active jobs. System operators may also use
324 such a program, in which case it would be running for a long
325 period of time and may also be interested in the jobs that have
326 completed. Finally such a program may be used to provide an
327 enhanced console and logging capability.
- 328 3. Collect resource usage for accounting or system utilization
329 purposes that copy the completed job statistics to an
330 accounting system. It is recognized that depending on
331 accounting programs to copy MIB data during the job-retention
332 period is somewhat unreliable, since the accounting program may
333 not be running (or may have crashed). Such a program is also
334 expected to keep a shadow copy of the entire Job Attribute
335 table including completed, canceled, and aborted jobs which the
336 program updates on each polling cycle. Such a program polls at
337 the rate of the persistence of the Attribute table. The design
338 is not optimized to help such an application determine which
339 jobs are completed, canceled, or aborted. Instead, the
340 application SHALL query each job that the application's shadow
341 copy shows was not complete, canceled, or aborted at the
342 previous poll cycle to see if it is now complete or canceled,
343 plus any new jobs that have been submitted.

344 The MIB provides a set of objects that represent a compatible subset of
345 job and document attributes of the ISO DPA standard[iso-dpa] and the
346 Internet Printing Protocol (IPP)[ipp-model], so that coherence is
347 maintained between these two protocols and the information presented to
348 end users and system operators by monitoring applications. However,
349 the job monitoring MIB is intended to be used with printers that
350 implement other job submitting and management protocols, such as IEEE
351 1284.1 (TIPSI)[tipsi], as well as with ones that do implement ISO DPA.

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

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

357 2. Terminology and Job Model

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

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

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

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

373 Attribute: A name, value-pair that specifies a job or document
374 instruction, a status, or a condition of a job or a document that has
375 been submitted to a server or device. A particular attribute NEED NOT
376 be present in each job instance. In other words, attributes are
377 present in a job instance only when there is a need to express the
378 value, either because (1) the client supplied a value in the job
379 submission protocol, (2) the document data contained an embedded
380 attribute, or (3) the server or device supplied a default value. An
381 agent SHALL represent an attribute as an entry (row) in the Attribute
382 table in this MIB in which entries are present only when necessary.
383 Attributes are identified in this MIB by an enum.

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

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

- 394 Document: A sub-section within a job that contains print data and
395 *document instructions* that apply to just the document.
- 396 Document Instruction: An instruction specifying how to process the
397 document. Document instructions MAY be passed in the job submission
398 protocol separate from the actual document data, or MAY be embedded in
399 the document data or a combination, depending on the job submission
400 protocol and implementation.
- 401 End User: A user that uses a client to submit a print job. See
402 "user".
- 403 Impression: For a print job, an impression is the passage of the
404 entire side of a sheet by the marker, whether or not any marks are made
405 and independent of the number of passes that the side makes past the
406 marker. Thus a four pass color process counts as a single impression,
407 as does highlight color. Impression counters count all kinds:
408 monochrome, highlight color, and full process color, while full color
409 counters only count full color impressions, and high light color
410 counters only count high light color impressions.
- 411 One-sided processing involves one impression per sheet. Two-sided
412 processing involves two impressions per sheet. If a two-sided document
413 has an odd number of pages, the last sheet still counts as two
414 impressions, if that sheet makes two passes through the marker or the
415 marker marks on both sides of a sheet in a single pass. Two-up
416 printing is the placement of two logical pages on one side of a sheet
417 and so is still a single impression. See "page" and "sheet".
- 418 NOTE - Since impressions include blank sides, it is suggested that
419 accounting application implementers consider charging for sheets,
420 rather than impressions, possibly using the value of the sides
421 attribute to select different charges for one-sided versus two-sided
422 printing, since some users may think that impressions don't include
423 blank sides.
- 424 Internal Collation: The production of the sheets for each document copy
425 performed within the printing device by making multiple passes over
426 either the source or an intermediate representation of the document.
- 427 Job: A unit of work whose results are expected together without
428 interjection of unrelated results. A job contains one or more
429 *documents*.
- 430 Job Accounting: The activity of a management application of accessing
431 the MIB and recording what happens to the job during and after the
432 processing of the job.

433 Job Instruction: An instruction specifying how, when, or where the job
434 is to be processed. Job instructions MAY be passed in the job
435 submission protocol or MAY be embedded in the document data or a
436 combination depending on the job submission protocol and
437 implementation.

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

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

446 Job Set: A group of jobs that are queued and scheduled together
447 according to a specified scheduling algorithm for a specified device or
448 set of devices. For implementations that embed the SNMP agent in the
449 device, the MIB job set normally represents *all* the jobs known to the
450 device, so that the implementation only implements a single job set.
451 If the SNMP agent is implemented in a server that controls one or more
452 devices, each MIB job set represents a job queue for (1) a specific
453 device or (2) set of devices, if the server uses a single queue to load
454 balance between several devices. Each job set is disjoint; no job
455 SHALL be represented in more than one MIB job set.

456 Monitor: Short for Job Monitoring Application.

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

460 Proxy: An agent that acts as a concentrator for one or more other
461 agents by accepting SNMP operations on the behalf of one or more other
462 agents, forwarding them on to those other agents, gathering responses
463 from those other agents and returning them to the original requesting
464 monitor.

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

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

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

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

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

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

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

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

484 Supervisor: A server that contains a control program that controls a
485 printer or other device. A supervisor is a client to the printer or
486 other device.

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

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

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

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

493 2.1 System Configurations for the Job Monitoring MIB

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

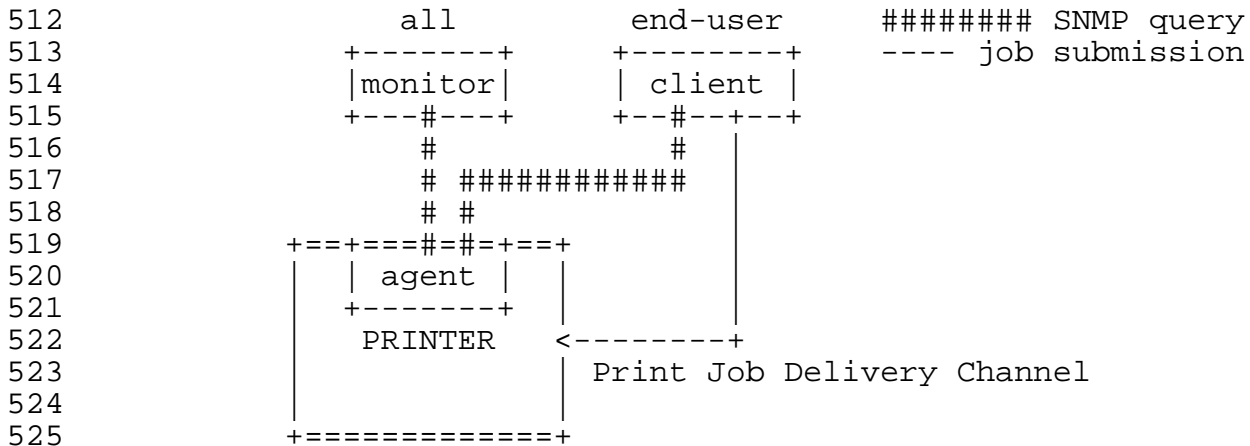
498 The diagram in the Printer MIB[print-mib] entitled: "One Printer's View
499 of the Network" is assumed for this MIB as well. Please refer to that
500 diagram to aid in understanding the following system configurations.

501 2.1.1 Configuration 1 - client-printer

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

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

511



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

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

- 529 1. Multiple clients MAY submit jobs to a printer.
- 530 2. Multiple clients MAY monitor a printer.
- 531 3. Multiple monitors MAY monitor a printer.
- 532 4. A client MAY submit jobs to multiple printers.
- 533 5. A monitor MAY monitor multiple printers.

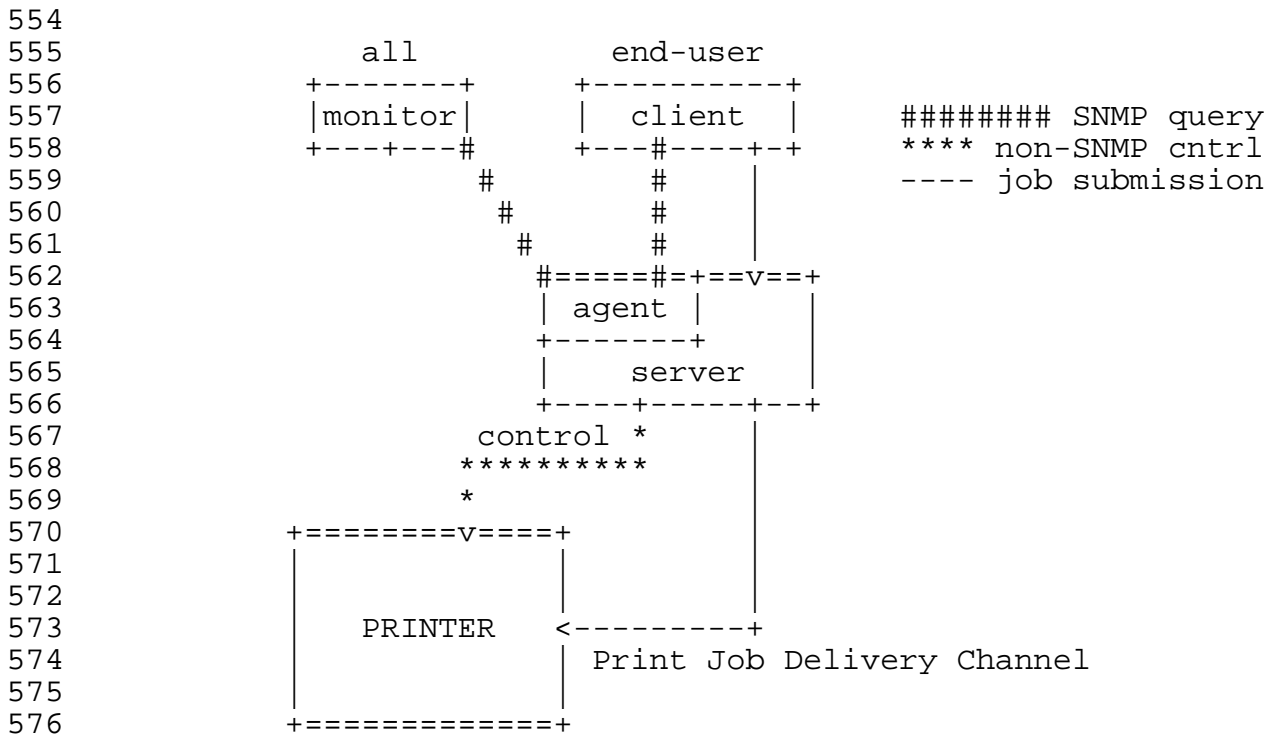
534 2.1.2 Configuration 2 - client-server-printer - agent in the server

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

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

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

543 There is no SNMP Job Monitoring MIB agent in the printer in
544 configuration 2, at least that the client or monitor are aware. In
545 this configuration, the agent SHALL return the current values of the
546 objects in the Job Monitoring MIB both for jobs the server keeps and
547 jobs that the server has submitted to the printer. The Job Monitoring
548 MIB agent SHALL obtain the required information from the printer by a
549 method that is beyond the scope of this document. The agent in the
550 server SHALL keep the job in the Job Monitoring MIB in the server as
551 long as the job is in the printer, plus a defined time period after the
552 job enters the completed state in which accounting programs can copy
553 out the accounting data from the Job Monitoring MIB.



577 Figure 2-2 - Configuration 2 - client-server-printer - agent in the
 578 server

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

- 581 1. Multiple clients MAY submit jobs to a server.
- 582 2. Multiple clients MAY monitor a server.
- 583 3. Multiple monitors MAY monitor a server.
- 584 4. A client MAY submit jobs to multiple servers.
- 585 5. A monitor MAY monitor multiple servers.
- 586 6. Multiple servers MAY submit jobs to a printer.
- 587 7. Multiple servers MAY control a printer.

588 2.1.3 Configuration 3 - client-server-printer - client monitors printer
 589 agent and server

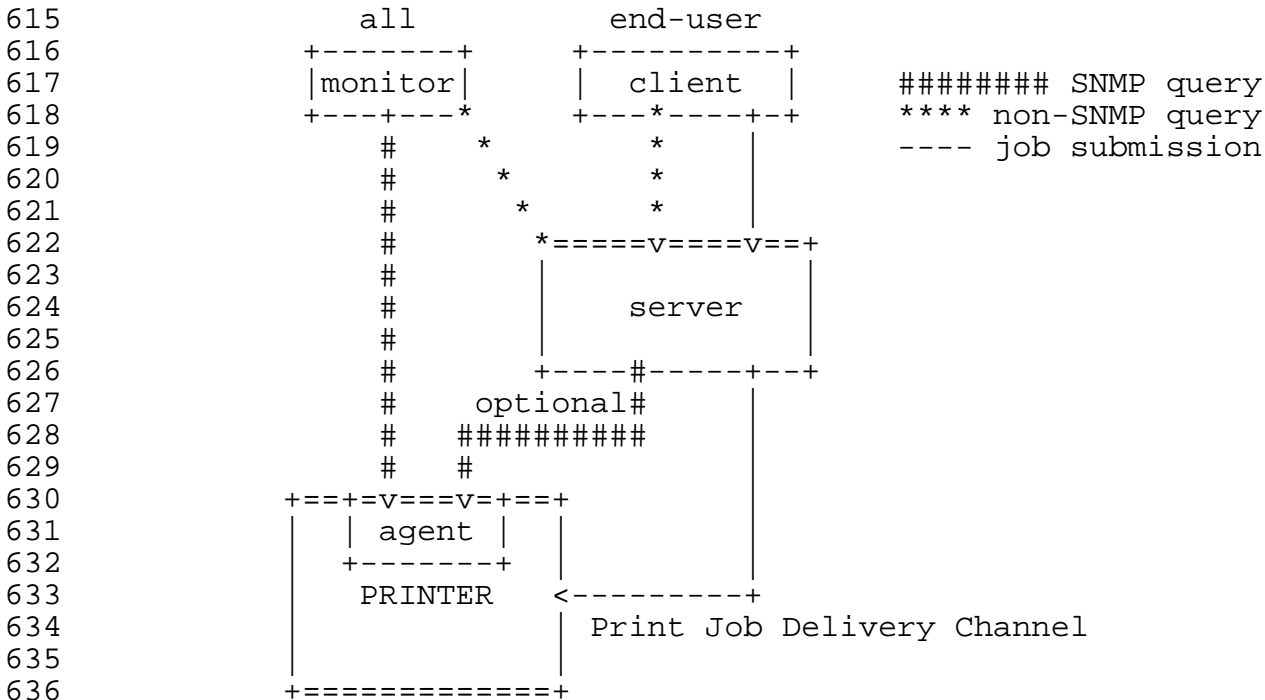
590 In the client-server-printer configuration 3, the client(s) submit jobs
 591 to an intermediate server by some network connection, *not* directly to
 592 the printer. That server does *not* contain a Job Monitoring MIB agent.

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

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

600 from the server after submitting the job to the printer usually
 601 almost immediately (before the job does much processing, if
 602 any).

603 In configuration 3, the agent (in the printer) SHALL keep the values of
 604 the objects in the Job Monitoring MIB that the agent implements updated
 605 for a job that the server has submitted to the printer. The agent
 606 SHALL obtain information about the jobs submitted to the printer from
 607 the server (either in the job submission protocol, in the document
 608 data, or by direct query of the server), in order to populate some of
 609 the objects the Job Monitoring MIB in the printer. The agent in the
 610 printer SHALL keep the job in the Job Monitoring MIB as long as the job
 611 is in the Printer, and longer in order to implement the completed state
 612 in which monitoring programs can copy out the accounting data from the
 613 Job Monitoring MIB.



637 Figure 2-3 - Configuration 3 - client-server-printer - client monitors
 638 printer agent and server

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

- 641 1. Multiple clients MAY submit jobs to a server.
- 642 2. Multiple clients MAY monitor a server.
- 643 3. Multiple monitors MAY monitor a server.
- 644 4. A client MAY submit jobs to multiple servers.
- 645 5. A monitor MAY monitor multiple servers.
- 646 6. Multiple servers MAY submit jobs to a printer.
- 647 7. Multiple servers MAY control a printer.

648 3. Managed Object Usage

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

650 3.1 Conformance Considerations

651 In order to achieve interoperability between job monitoring
652 applications and job monitoring agents, this specification includes the
653 conformance requirements for both monitoring applications and agents.

654 3.1.1 Conformance Terminology

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

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

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

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

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

670 3.1.2 Agent Conformance Requirements

671 A conforming agent:

- 672 1. SHALL implement *all* MANDATORY groups in this specification.
- 673 2. SHALL implement any attributes if (1) the server or device
674 supports the functionality represented by the attribute and (2)
675 the information is available to the agent.
- 676 3. SHOULD implement both forms of an attribute if it implements an
677 attribute that permits a choice of INTEGER and OCTET STRING
678 forms, since implementing both forms may help management
679 applications by giving them a choice of representations, since
680 the representation are equivalent. See the JmAttributeTypeTC
681 textual-convention.

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

684 3.1.2.1 MIB II System Group objects

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

688 3.1.2.2 MIB II Interface Group objects

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

692 3.1.2.3 Printer MIB objects

693 If the agent is providing access to a device that is a printer, the
694 agent SHALL implement all of the MANDATORY objects in the Printer
695 MIB[print-mib] and all the objects in other MIBs that conformance to
696 the Printer MIB requires, such as the Host Resources MIB[hr-mib]. If
697 the agent is providing access to a server that controls one or more
698 direct-connect or networked printers, the agent NEED NOT implement the
699 Printer MIB and NEED NOT implement the Host Resources MIB.

700 3.1.3 Job Monitoring Application Conformance Requirements

701 A conforming job monitoring application:

- 702 1. SHALL accept the full syntactic range for all objects in all
703 MANDATORY groups and all MANDATORY attributes that are required
704 to be implemented by an agent according to Section 3.1.2 and
705 SHALL either present them to the user or ignore them.
- 706 2. SHALL accept the full syntactic range for *all* attributes,
707 including enum and bit values specified in this specification
708 and additional ones that may be registered with the PWG and
709 SHALL either present them to the user or ignore them. In
710 particular, a conforming job monitoring application SHALL not
711 malfunction when receiving any standard or registered enum or
712 bit values. See Section 3.7 entitled "IANA and PWG
713 Registration Considerations".
- 714 3. SHALL NOT fail when operating with agents that materialize
715 attributes *after* the job has been submitted, as opposed to when
716 the job is submitted.
- 717 4. SHALL, if it supports a time attribute, accept either form of
718 the time attribute, since agents are free to implement either
719 time form.

720 3.2 The Job Tables and the Oldest Active and Newest Active Indexes

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

- 723 1. jmGeneralJobSetIndex - which job set
- 724 2. jmJobIndex - which job in the job set

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

- 728 1. jmGeneralOldestActiveJobIndex - the index of the active job
729 that has been in the tables the longest.
- 730 2. jmGeneralNewestActiveJobIndex - the index of the active job
731 that has been most recently added to the tables.

732 The agent SHALL assign the next incremental value of jmJobIndex to the
733 job, when a new job is accepted by the server or device to which the
734 agent is providing access. If the incremented value of jmJobIndex
735 would exceed the implementation-defined maximum value for jmJobIndex,
736 the agent SHALL 'wrap' back to 1. An agent uses the resulting value of
737 jmJobIndex for storing information in the jmJobTable and the
738 jmAttributeTable about the job.

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

744 It is recommended that agents that are providing access to
745 servers/devices that already allocate job-identifiers for jobs as
746 integers use the same integer value for the jmJobIndex. Then
747 management applications using this MIB and applications using other
748 protocols will see the same job identifiers for the same jobs. Agents
749 providing access to systems that contain jobs with a job identifier of
750 0 SHALL map the job identifier value 0 to a jmJobIndex value that is
751 one higher than the highest job identifier value that any job can have
752 on that system. Then only job 0 will have a different job-identifier
753 value than the job's jmJobIndex value.

754 NOTE - If a server or device accepts jobs using multiple job submission
755 protocols, it may be difficult for the agent to meet the recommendation
756 to use the job-identifier values that the server or device assigns as
757 the jmJobIndex value, unless the server/device assigns job-identifiers
758 for each of its job submission protocols from the same job-identifier
759 number space.

760 Each time a new job is accepted by the server or device that the agent
761 is providing access to AND that job is to be 'active' (pending,
762 processing, or processingStopped, but not pendingHeld), the agent SHALL
763 copy the value of the job's jmJobIndex to the
764 jmGeneralNewestActiveJobIndex object. If the new job is to be
765 'inactive' (pendingHeld state), the agent SHALL not change the value of
766 jmGeneralNewestActiveJobIndex object (though the agent SHALL assign the
767 next incremental jmJobIndex value to the job).

768 When a job transitions from one of the 'active' job states (pending,
769 processing, processingStopped) to one of the 'inactive' job states
770 (pendingHeld, completed, canceled, or aborted), with a jmJobIndex value
771 that matches the jmGeneralOldestActiveJobIndex object, the agent SHALL
772 advance (or wrap) the value to the next oldest 'active' job, if any.
773 See the JmJobStateTC textual-convention for a definition of the job
774 states.

775 Whenever a job transitions from one of the 'inactive' job states to one
776 of the 'active' job states (from pendingHeld to pending or processing),
777 the agent SHALL update the value of either the
778 jmGeneralOldestActiveJobIndex or the jmGeneralNewestActiveJobIndex
779 objects, or both, if the job's jmJobIndex value is outside the range
780 between jmGeneralOldestActiveJobIndex and
781 jmGeneralNewestActiveJobIndex.

782 When all jobs become 'inactive', i.e., enter the pendingHeld,
783 completed, canceled, or aborted states, the agent SHALL set the value
784 of both the jmGeneralOldestActiveJobIndex and
785 jmGeneralNewestActiveJobIndex objects to 0.

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

791 If an application detects that the jmGeneralNewestActiveJobIndex is
792 smaller than jmGeneralOldestActiveJobIndex, the job index has wrapped.
793 In this case, the application SHALL reset the index to 1 when the end
794 of the table is reached and continue the GetNext operations to find the
795 rest of the active jobs.

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

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

803 3.3 The Attribute Mechanism

804 Attributes are similar to information objects, except that attributes
805 are identified by an enum, instead of an OID, so that attributes may be
806 registered without requiring a new MIB. Also an implementation that
807 does not have the functionality represented by the attribute can omit
808 the attribute entirely, rather than having to return a distinguished
809 value. The agent is free to materialize an attribute in the
810 jmAttributeTable as soon as the agent is aware of the value of the
811 attribute.

812 The agent materializes job attributes in a four-indexed
813 jmAttributeTable:

- 814 1. jmGeneralJobSetIndex - which job set
- 815 2. jmJobIndex - which job in the job set
- 816 3. jmAttributeTypeIndex - which attribute
- 817 4. jmAttributeInstanceIndex - which attribute instance for those
818 attributes that can have multiple values per job.

819 Some attributes represent information about a job, such as a file-name,
820 a document-name, a submission-time or a completion time. Other
821 attributes represent resources required, e.g., a medium or a colorant,
822 etc. to process the job before the job starts processing OR to indicate
823 the amount of the resource consumed during and after processing, e.g.,
824 pages completed or impressions completed. If both a required and a
825 consumed value of a resource is needed, this specification assigns two
826 separate attribute enums in the textual convention.

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

834 3.3.1 Conformance of Attribute Implementation

835 An agent SHALL implement any attribute if (1) the server or device
836 supports the functionality represented by the attribute and (2) the
837 information is available to the agent. The agent MAY create the
838 attribute row in the jmAttributeTable when the information is available
839 or MAY create the row earlier with the designated 'unknown' value
840 appropriate for that attribute. See next section.

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

844 3.3.2 Useful, 'Unknown', and 'Other' Values for Objects and Attributes

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

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

854 SHALL be generated by the agent. This MIB has also been designed so
855 that when an agent materializes an attribute, the agent SHALL
856 materialize a row consisting of both the jmAttributeValueAsInteger and
857 jmAttributeValueAsOctets objects.

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

865 Since each attribute is represented by a row consisting of both the
866 jmAttributeValueAsInteger and jmAttributeValueAsOctets MANDATORY
867 objects, SNMP requires that the agent SHALL always create an attribute
868 row with both objects specified. However, for most attributes the
869 agent SHALL return a "useful" value for one of the objects and SHALL
870 return the 'other' value for the other object. For integer only
871 attributes, the agent SHALL always return a zero-length string value
872 for the jmAttributeValueAsOctets object. For octet string only
873 attributes, the agent SHALL always return a '-1' value for the
874 jmAttributeValueAsInteger object.

875 3.3.3 Data Sub-types and Attribute Naming Conventions

876 Many attributes are sub-typed to give a more specific data type than
877 Integer32 or OCTET STRING. The data sub-type of each attribute is
878 indicated on the first line(s) of the description. Some attributes
879 have several different data sub-type representations. When an
880 attribute has both an Integer32 data sub-type and an OCTET STRING data
881 sub-type, the attribute can be represented in a single row in the
882 jmAttributeTable. In this case, the data sub-type name is not included
883 as the last part of the name of the attribute, e.g., documentFormat(38)
884 which is both an enum and/or a name. When the data sub-types cannot be
885 represented by a single row in the jmAttributeTable, each such
886 representation is considered a separate attribute and is assigned a
887 separate name and enum value. For these attributes, the name of the
888 data sub-type is the last part of the name of the attribute: Name,
889 Index, DateAndTime, TimeStamp, etc. For example,
890 documentFormatIndex(37) is an index.

891 NOTE: The Table of Contents also lists the data sub-type and/or data
892 sub-types of each attribute, using the textual-convention name when
893 such is defined. The following abbreviations are used in the Table of
894 Contents as shown:
895

| | |
|----------------|--|
| '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. |

896 3.3.4 Single-Value (Row) Versus Multi-Value (MULTI-ROW) Attributes

897 Most attributes SHALL have only one row per job. However, a few
898 attributes can have multiple values per job or even per document, where
899 each value is a separate row in the jmAttributeTable. Unless indicated
900 with 'MULTI-ROW:' in the JmAttributeTypeTC description, an agent SHALL
901 ensure that each attribute occurs only once in the jmAttributeTable for
902 a job. Most of the 'MULTI-ROW' attributes do not allow duplicate
903 values, i.e., the agent SHALL ensure that each value occurs only once
904 for a job. Only if the specification of the 'MULTI-ROW' attribute also
905 says "the values NEED NOT be unique" can the agent allow duplicate
906 values to occur for the job.

907 NOTE - Duplicates are allowed for 'extensive' 'MULTI-ROW' attributes,
908 such as fileName(34) or documentName(35) which are specified to be
909 'per-document' attributes, but are *not* allowed for 'intensive' 'MULTI-
910 ROW' attributes, such as mediumConsumed(171) and documentFormat(38)
911 which are specified to be 'per-job' attributes.

912 3.3.5 Requested Objects and Attributes

913 A number of objects and attributes record requirements for the job.
914 Such object and attribute names end with the word 'Requested'. In the
915 interests of brevity, the phrase 'requested' SHALL mean: (1) requested
916 by the client (or intervening server) in the job submission protocol
917 and MAY also mean (2) embedded in the submitted document data, and/or
918 (3) defaulted by the recipient device or server with the same semantics
919 as if the requester had supplied, depending on implementation. Also if
920 a value is supplied by the job submission client, and the server/device
921 determines a better value, through processing or other means, the agent
922 MAY return that better value for such object and attribute.

923 3.3.6 Consumption Attributes

924 A number of objects and attributes record consumption. Such attribute
925 names end with the word 'Completed' or 'Consumed'. If the job has not
926 yet consumed what that resource is metering, the agent either: (1)
927 SHALL return the value 0 or (2) SHALL not add this attribute to the
928 jmAttributeTable until the consumption begins. In the interests of
929 brevity, the semantics for 0 is specified once here and is not repeated
930 for each consumption attribute specification and a DEFVAL of 0 is
931 indicated.

932 3.3.7 Index Value Attributes

933 A number of attributes are indexes in other tables. Such attribute
934 names end with the word 'Index'. If the agent has not (yet) assigned
935 an index value for a particular index attribute for a job, the agent
936 SHALL either: (1) return the value 0 or (2) not add this attribute to
937 the jmAttributeTable until the index value is assigned. In the
938 interests of brevity, the semantics for 0 is specified once here and is
939 not repeated for each index attribute specification and a DEFVAL of 0
940 is indicated.

941 3.4 Monitoring Job Progress

942 There are a number of objects and attributes for monitoring the
943 progress of a job. These objects and attributes count the number of K
944 octets, impressions, sheets, and pages requested or completed. For
945 impressions and sheets, "completed" SHALL mean stacked, unless the
946 implementation is unable to detect when each sheet is stacked, in which
947 case stacked is approximated when processing of each sheet completes.
948 There are objects and attributes for the overall job and for the
949 current copy of the document currently being stacked. For the latter,
950 the rate at which the various objects and attributes count depends on
951 the sheet and document collation of the job.

952 Job Collation included sheet collation and document collation. Sheet
953 collation is defined to be the ordering of sheets within a document
954 copy. Document collation is defined to be ordering of document copies
955 within a multi-document job. There are three types of job collation
956 (see terminology definitions in Section 2):

- 957 1. Uncollated Sheets - No collation of the sheets within each
958 document copy, i.e., each sheet of a document that is to
959 produce multiple copies is replicated before the next sheet in
960 the document is processed and stacked. If the device has an
961 output bin collator, uncollated sheets may actually produce
962 collated sheets as far as the user is concerned (in the output
963 bins). However, when the job collation is 'uncollated sheets',
964 job progress is indistinguishable to a monitoring application
965 between a device that has an output bin collator and one that
966 does not.

967 2. Collated Documents - Collation of the sheets within each
968 document copy is performed within the printing device by making
969 multiple passes over either the source or an intermediate
970 representation of the document. In addition, when there are
971 multiple documents per job, the i'th copy of each document is
972 stacked before the j'th copy of each document, i.e., the
973 documents are collated within each job copy. For example, if a
974 job is submitted with documents, A and B, the job is made
975 available to the end user as: A, B, A, B, ... Collated Document
976 correspond to the IPP [ipp-model] 'separate-documents-collated-
977 copies' value of the "multiple-document-handling" attribute.

978
979 If jobCopiesRequested or documentCopiesRequested = 1, then
980 jobCollationType is defined as 4.

981 3. Uncollated Documents - Collation of the sheets within each
982 document copy is performed within the printing device by making
983 multiple passes over either the source or an intermediate
984 representation of the document. In addition, when there are
985 multiple documents per job, all copies of the first document in
986 the job are stacked before the any copied of the next document
987 in the job, i.e., the documents are uncollated within the job.
988 For example, if a job is submitted with documents, A and B, the
989 job is mad available to the end user as: A, A, ..., B, B, ...
990 Uncollated Documents correspond to the IPP [ipp-model]
991 'separate-documents-uncollated-copies' value of the "multiple-
992 document-handling" attribute.

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

995 1. jmJobImpressionsCompleted - counts the total number of
996 impressions stacked for the job

997 2. impressionsCompletedCurrentCopy - counts the number of
998 impressions stacked for the current document copy

999 3. sheetCompletedCopyNumber - identifies the number of the copy
1000 for the current document being stacked where the first copy is
1001 1.

1002 4. sheetCompletedDocumentNumber - identifies the current document
1003 within the job that is being stacked where the first document
1004 in a job is 1. NOTE: this attribute SHOULD NOT be implemented
1005 for implementations that only support one document per job.

1006 For each of the three types of job collation, a job with three copies
1007 of two documents (1, 2), where each document consists of 3 impressions,
1008 the four variables have the following values as each sheet is stacked
1009 for one-sided printing:

1010 Job Collation Type = Uncollated Sheets

1011

| 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 |

1012

1013 Job Collation Type = Collated Documents

1014

| 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 |

1015

1016 Job Collation Type = Uncollated Documents
 1017

| 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 |

1018

1019 3.5 Job Identification

1020 There are a number of attributes that permit a user, operator or system
 1021 administrator to identify jobs of interest, such as jobURI, jobName,
 1022 jobOriginatingHost, etc. In addition, there is a jmJobSubmissionID
 1023 object that is a text string table index. Being a table index allows a
 1024 monitoring application to quickly locate and identify a particular job
 1025 of interest that was submitted from a particular client by the user
 1026 invoking the monitoring application without having to scan the entire
 1027 job table. The Job Monitoring MIB needs to provide for identification
 1028 of the job at both sides of the job submission process. The primary
 1029 identification point is the client side. The jmJobSubmissionID allows
 1030 the monitoring application to identify the job of interest from all the
 1031 jobs currently "known" by the server or device. The value of
 1032 jmJobSubmissionID can be assigned by either the client's local system
 1033 or a downstream server or device. The point of assignment depends on
 1034 the job submission protocol in use.

1035 The server/device-side identifier, called the jmJobIndex object, SHALL
 1036 be assigned by the SNMP Job Monitoring MIB agent when the server or
 1037 device accepts the jobs from submitting clients. The jmJobIndex object
 1038 allows the interested party to obtain all objects desired that relate
 1039 to a particular job. See Section 3.2, entitled 'The Job Tables and the

1040 Oldest Active and Newest Active Indexes' for the specification of how
1041 the agent SHALL assign the jmJobIndex values.

1042 The MIB provides a mapping table that maps each jmJobSubmissionID value
1043 to a corresponding jmJobIndex value generated by the agent, so that an
1044 application can determine the correct value for the jmJobIndex value
1045 for the job of interest in a single Get operation, given the Job
1046 Submission ID. See the jmJobIDGroup.

1047 In some configurations there may be more than one application program
1048 that monitors the same job when the job passes from one network entity
1049 to another when it is submitted. See configuration 3. When there are
1050 multiple job submission IDs, each entity MAY supply an appropriate
1051 jmJobSubmissionID value. In this case there would be a separate entry
1052 in the jmJobSubmissionID table, one for each jmJobSubmissionID. All
1053 entries would map to the same jmJobIndex that contains the job data.
1054 When the job is deleted, it is up to the agent to remove all entries
1055 that point to the job from the jmJobSubmissionID table as well.

1056 The jobName attribute provides a name that the user supplies as a job
1057 attribute with the job. The jobName attribute is not necessarily
1058 unique, even for one user, let alone across users.

1059 3.6 Internationalization Considerations

1060 This section describes the internationalization considerations included
1061 in this MIB.

1062 3.6.1 Text generated by the server or device

1063 There are a few objects and attributes generated by the server or
1064 device that SHALL be represented using the Universal Multiple-Octet
1065 Coded Character Set (UCS) [ISO-10646]. These objects and attributes
1066 are always supplied (if implemented) by the agent, not by the job
1067 submitting client:

- 1068 1. jmGeneralJobSetName object
- 1069 2. processingMessage(6) attribute
- 1070 3. physicalDevice(32) (name value) attribute

1071 The character encoding scheme for representing these objects and
1072 attributes SHALL be UTF-8 as recommended by RFC 2130 [RFC 2130] and the
1073 "IETF Policy on Character Sets and Language" [char-set policy]. The
1074 'JmUTF8StringTC' textual convention is used to indicate UTF-8 text
1075 strings.

1076 NOTE - For strings in 7-bit US-ASCII, there is no impact since the UTF-
1077 8 representation of 7-bit ASCII is identical to the US-ASCII [US-ASCII]
1078 encoding.

1079 The text contained in the processingMessage(6) attribute is generated
1080 by the server/device. The natural language for the
1081 processingMessage(6) attribute is identified by the

1082 processingMessageNaturalLangTag(7) attribute. The
1083 processingMessageNaturalLangTag(7) attribute uses the
1084 JmNaturalLanguageTagTC textual convention which SHALL conform to the
1085 language tag mechanism specified in RFC 1766 [RFC-1766]. The
1086 JmNaturalLanguageTagTC value is the same as the IPP [IPP-model]
1087 'naturalLanguage' attribute syntax. RFC 1766 specifies that a US-ASCII
1088 string consisting of the natural language followed by an optional
1089 country field. Both fields use the same two-character codes from ISO
1090 639 [ISO-639] and ISO 3166 [ISO-3166], respectively, that are used in
1091 the Printer MIB for identifying language and country.

1092 Examples of the values of the processingMessageNaturalLangTag(7)
1093 attribute include:

- 1094 1. 'en' for English
- 1095 2. 'en-us' for US English
- 1096 3. 'fr' for French
- 1097 4. 'de' for German

1098 3.6.2 Text supplied by the job submitter

1099 All of the objects and attributes represented by the 'JmJobStringTC'
1100 textual-convention are either (1) supplied in the job submission
1101 protocol by the client that submits the job to the server or device or
1102 (2) are defaulted by the server or device if the job submitting client
1103 does not supply values. The agent SHALL represent these objects and
1104 attributes in the MIB either (1) in the coded character set as they
1105 were submitted or (2) MAY convert the coded character set to another
1106 coded character set or encoding scheme. In any case, the resulting
1107 coded character set representation SHOULD be UTF-8 [UTF-8], but SHALL
1108 be one in which the code positions from 0 to 31 SHALL not be used, 32
1109 to 127 SHALL be US-ASCII [US-ASCII], 127 SHALL be unused, and the
1110 remaining code positions 128 to 255 SHALL represent single-byte or
1111 multi-byte graphic characters structured according to ISO 2022 [ISO
1112 2022] or SHALL be unused.

1113 The coded character set SHALL be one of the ones registered with IANA
1114 [IANA] and SHALL be identified by the jobCodedCharSet attribute in the
1115 jmJobAttributeTable for the job. If the agent does not know what coded
1116 character set was used by the job submitting client, the agent SHALL
1117 either (1) return the 'unknown(2)' value for the jobCodedCharSet
1118 attribute or (2) not return the jobCodedCharSet attribute for the job.

1119 Examples of coded character sets which meet this criteria for use as
1120 the value of the jobCodedCharSet job attribute are: US-ASCII [US-
1121 ASCII], ISO 8859-1 (Latin-1) [ISO 8859-1], any ISO 8859-n, HP Roman8,
1122 IBM Code Page 850, Windows Default 8-bit set, UTF-8 [UTF-8], US-ASCII
1123 plus JIS X0208-1990 Japanese [JIS X0208], US-ASCII plus GB2312-1980 PRC
1124 Chinese [GB2312]. See the IANA registry of coded character sets [IANA
1125 charsets].

1126 Examples of coded character sets which do not meet this criteria are:
1127 national 7-bit sets conforming to ISO 646 (except US-ASCII), EBCDIC,

1128 and ISO 10646 (Unicode) [ISO-10646]. In order to represent Unicode
1129 characters, the UTF-8 [UTF-8] encoding scheme SHALL be used which has
1130 been assigned the MIBenum value of '106' by IANA.

1131 The jobCodedCharSet attribute uses the imported 'CodedCharSet' textual-
1132 convention from the Printer MIB [printmib].

1133 The natural language for attributes represented by the textual-
1134 convention JmJobStringTC SHALL be identified either (1) by the
1135 jobNaturalLanguageTag(9) attribute or SHALL be keywords in US-English
1136 (as in IPP). A monitoring application SHOULD attempt to localize
1137 keywords into the language of the user by means of some lookup
1138 mechanism. If the keyword value is not known to the monitoring
1139 application, the monitoring application SHOULD assume that the value is
1140 in the natural language specified by the job's jobNaturalLanguageTag(9)
1141 attribute and SHOULD present the value to its user as is. The
1142 jobNaturalLanguageTag(9) attribute value SHALL have the same syntax and
1143 semantics as the processingMessageNaturalLangTag(7) attribute, except
1144 that the jobNaturalLanguageTag(9) attribute identifies the natural
1145 language of attributes supplied by the job submitter instead of the
1146 natural language of the processingMessage(6) attribute. See Section
1147 3.6.1.

1148 3.6.3 'DateAndTime' for representing the date and time

1149 This MIB also contains objects that are represented using the
1150 DateAndTime textual convention from SMIV2 [SMIV2-TC]. The job
1151 management application SHALL display such objects in the locale of the
1152 user running the monitoring application.

1153 3.7 IANA and PWG Registration Considerations

1154 This MIB does not require any additional registration schemes for IANA,
1155 but does depend on registration schemes that other Internet standards
1156 track specifications have set up. The names of these IANA registration
1157 assignments under the /in-notes/iana/assignments/ path:

- 1158 1. printer-language-numbers - used as enums in the documentFormat(38)
1159 attribute
- 1160 2. media-types - uses as keywords in the documentFormat(38) attribute
- 1161 3. character-sets - used as enums in the jobCodedCharSet(8) attribute

1162 During the development of this standard, the Printer Working Group
1163 (PWG) will register additional enums while the standard is in the
1164 proposed and draft states according to the procedures described in this
1165 section. The PWG will handle registration of additional enums after
1166 approving this standard, according to the procedures described in this
1167 section:

1168 3.7.1 PWG Registration of enums

1169 This specification uses textual conventions to define enumerated values
1170 (enums) and bit values. Enumerations (enums) and bit values are sets
1171 of symbolic values defined for use with one or more objects or
1172 attributes. All enumeration sets and bit value sets are assigned a
1173 symbolic data type name (textual convention). As a convention the
1174 symbolic name ends in "TC" for textual convention. These enumerations
1175 are defined at the beginning of the MIB module specification.

1176 The PWG has defined several type of enumerations for use in the Job
1177 Monitoring MIB and the Printer MIB[print-mib]. These types differ in
1178 the method employed to control the addition of new enumerations.
1179 Throughout this document, references to "type n enum", where n can be
1180 1, 2 or 3 can be found in the various tables. The definitions of these
1181 types of enumerations are:

1182 3.7.1.1 Type 1 enumerations

1183 Type 1 enumeration: All the values are defined in the Job Monitoring
1184 MIB specification (RFC for the Job Monitoring MIB). Additional
1185 enumerated values require a new RFC.

1186 There are no type 1 enums in the current draft.

1187 3.7.1.2 Type 2 enumerations

1188 Type 2 enumeration: An initial set of values are defined in the Job
1189 Monitoring MIB specification. Additional enumerated values are
1190 registered with the PWG.

1191 The following type 2 enums are contained in the current draft :

- 1192 1. JmUTF8StringTC
- 1193 2. JmJobStringTC
- 1194 3. JmNaturalLanguageTagTC
- 1195 4. JmTimeStampTC
- 1196 5. JmFinishingTC [same enum values as IPP "finishing" attribute]
- 1197 6. JmPrintQualityTC [same enum values as IPP "print-quality"
- 1198 attribute]
- 1199 7. JmTonerEconomyTC
- 1200 8. JmMediumTypeTC
- 1201 9. JmJobSubmissionIDTypeTC
- 1202 10. JmJobCollationTypeTC
- 1203 11. JmJobStateTC [same enum values as IPP "job-state" attribute]
- 1204 12. JmAttributeTypeTC

1205 For those textual conventions that have the same enum values as the
1206 indicated IPP Job attribute SHALL be simultaneously registered by the
1207 PWG for use with IPP [ipp-model] and the Job Monitoring MIB.

1208 3.7.1.3 Type 3 enumeration

1209 Type 3 enumeration: An initial set of values are defined in the Job
1210 Monitoring MIB specification. Additional enumerated values are
1211 registered through the PWG without PWG review.

1212 There are no type 3 enums in the current draft.

1213 3.7.2 PWG Registration of type 2 bit values

1214 This draft contains the following type 2 bit value textual-conventions:

- 1215 1. JmJobServiceTypesTC
- 1216 2. JmJobStateReasons1TC
- 1217 3. JmJobStateReasons2TC
- 1218 4. JmJobStateReasons3TC
- 1219 5. JmJobStateReasons4TC

1220 These textual-conventions are defined as bits in an Integer so that
1221 they can be used with SNMPv1 SMI. The jobStateReasonsN (N=1..4)
1222 attributes are defined as bit values using the corresponding
1223 JmJobStateReasonsMTC textual-conventions.

1224 The registration of JmJobServiceTypesTC and JmJobStateReasonsMTC bit
1225 values SHALL follow the procedures for a type 2 enum as specified in
1226 Section 3.7.1.2.

1227 3.7.3 PWG Registration of Job Submission Id Formats

1228 In addition to enums and bit values, this specification assigns a
1229 single ASCII digit or letter to various job submission ID formats. See
1230 the JmJobSubmissionIDTypeTC textual-convention and the object. The
1231 registration of JobSubmissionID format numbers SHALL follow the
1232 procedures for a type 2 enum as specified in Section 3.7.1.2.

1233 3.7.4 PWG Registration of MIME types/sub-types for document-formats

1234 The documentFormat(38) attribute has MIME type/sub-type values for
1235 indicating document formats which IANA registers as "media type" names.
1236 The values of the documentFormat(38) attribute are the same as the
1237 corresponding Internet Printing Protocol (IPP) "document-format" Job
1238 attribute values [ipp-model].

1239 3.8 Security Considerations

1240 3.8.1 Read-Write objects

1241 All objects are read-only, greatly simplifying the security
1242 considerations. If another MIB augments this MIB, that MIB might
1243 accept SNMP Write operations to objects in that MIB whose effect is to
1244 modify the values of read-only objects in this MIB. However, that MIB
1245 SHALL have to support the required access control in order to achieve
1246 security, not this MIB.

1247 3.8.2 Read-Only Objects In Other User's Jobs

1248 The security policy of some sites MAY be that unprivileged users can
1249 only get the objects from jobs that they submitted, plus a few minimal
1250 objects from other jobs, such as the jmJobKOctetsPerCopyRequested and
1251 jmJobKOctetsProcessed objects, so that a user can tell how busy a
1252 printer is. Other sites MAY allow all unprivileged users to see all
1253 objects of all jobs. This MIB does not require, nor does it specify
1254 how, such restrictions would be implemented. A monitoring application
1255 SHOULD enforce the site security policy with respect to returning
1256 information to an unprivileged end user that is using the monitoring
1257 application to monitor jobs that do not belong to that user, i.e., the
1258 jmJobOwner object in the jmJobTable does not match the user's user
1259 name.

1260 An operator is a privileged user that would be able to see all objects
1261 of all jobs, independent of the policy for unprivileged users.

1262 3.9 Notifications

1263 This MIB does not specify any notifications. For simplicity,
1264 management applications are expected to poll for status. The
1265 jmGeneralJobPersistence and jmGeneralAttributePersistence objects
1266 assist an application to determine the polling rate. The resulting
1267 network traffic is not expected to be significant.

1268 4. MIB specification

1269 The following pages constitute the actual Job Monitoring MIB.

```
1270 Job-Monitoring-MIB DEFINITIONS ::= BEGIN
1271
1272 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

1273
1274 -- Use the enterprises arc assigned to the PWG which is pwg(2699).
1275 -- Assign the first value: jobmonMIB(1) immediately under pwg(2669).
1276
1277 jobmonMIB MODULE-IDENTITY
1278     LAST-UPDATED "9801130000Z"
1279     ORGANIZATION "Printer Working Group (PWG)"
1280     CONTACT-INFO
1281         "Tom Hastings
1282         Postal: Xerox Corp.
1283                 Mail stop ESAE-231
1284                 701 S. Aviation Blvd.
1285                 El Segundo, CA 90245
1286
1287         Tel:      (301)333-6413
1288         Fax:      (301)333-5514
1289         E-mail:   hastings@cpl0.es.xerox.com
1290
1291         Send questions and comments to the Printer Working Group (PWG)
1292         using the Job Monitoring Project (JMP) Mailing List:
1293         jmp@pwg.org
1294
1295         For further information, including how to subscribe to the
1296         jmp mailing list, access the PWG web page under 'JMP':
1297
1298         http://www.pwg.org/
1299
1300         Implementers of this specification are encouraged to join the
1301         jmp mailing list in order to participate in discussions on any
1302         clarifications needed and registration proposals being reviewed
1303         in order to achieve consensus."
1304     DESCRIPTION
1305         "The MIB module for monitoring job in servers, printers, and
1306         other devices.
1307
1308         Version: 1.0"
1309     ::= { enterprises pwg(2699) jobmonMIB(1) }
```

```
1310
1311 -- Textual conventions for this MIB module
1312
1313 JmUTF8StringTC ::= TEXTUAL-CONVENTION
1314     DISPLAY-HINT "255a"
1315     STATUS      current
1316     DESCRIPTION
1317         "To facilitate internationalization, this TC represents
1318         information taken from the ISO/IEC IS 10646-1 character set,
1319         encoded as an octet string using the UTF-8 character encoding
1320         scheme."
1321     REFERENCE
1322         "See section 3.6.1, entitled: 'Text generated by the server or
1323         device'."
1324     SYNTAX      OCTET STRING (SIZE (0..63))
1325
1326
1327
1328
1329 JmJobStringTC ::= TEXTUAL-CONVENTION
1330     STATUS      current
1331     DESCRIPTION
1332         "To facilitate internationalization, this TC represents
1333         information using any coded character set registered by IANA as
1334         specified in section 3.7. While it is recommended that the
1335         coded character set be UTF-8 [UTF-8], the actual coded
1336         character set SHALL be indicated by the value of the
1337         jobCodedCharSet(8) attribute for the job."
1338     REFERENCE
1339         "See section 3.6.2, entitled: 'Text supplied by the job
1340         submitter'."
1341     SYNTAX      OCTET STRING (SIZE (0..63))
1342
1343
1344
1345
1346 JmNaturalLanguageTagTC ::= TEXTUAL-CONVENTION
1347     STATUS      current
1348     DESCRIPTION
1349         "An IETF RFC 1766-compliant 'language tag', with zero or more
1350         sub-tags that identify a natural language. While RFC 1766
1351         specifies that the US-ASCII values are case-insensitive, this
1352         MIB specification requires that all characters SHALL be lower
1353         case in order to simplify comparing by management
1354         applications."
1355     REFERENCE
1356         "See section 3.6.1, entitled: 'Text generated by the server or
1357         device' and section 3.6.2, entitled: 'Text supplied by the job
1358         submitter'."
1359     SYNTAX      OCTET STRING (SIZE (0..63))
1360
1361
```

```

1362 JmTimeStampTC ::= TEXTUAL-CONVENTION
1363     STATUS      current
1364     DESCRIPTION
1365         "The simple time at which an event took place.  The units SHALL
1366         be in seconds since the system was booted.
1367
1368         NOTE - JmTimeStampTC is defined in units of seconds, rather
1369         than 100ths of seconds, so as to be simpler for agents to
1370         implement (even if they have to implement the 100ths of a
1371         second to comply with implementing sysUpTime in MIB-II[mib-
1372         II].)
1373
1374         NOTE - JmTimeStampTC is defined as an Integer32 so that it can
1375         be used as a value of an attribute, i.e., as a value of the
1376         jmAttributeValueAsInteger object.  The TimeStamp textual-
1377         convention defined in SNMPv2-TC [SMIv2-TC] is defined as an
1378         APPLICATION 3 IMPLICIT INTEGER tag, not an Integer32 which is
1379         defined in SNMPv2-SMI [SMIv2-TC] as UNIVERSAL 2 IMPLICIT
1380         INTEGER, so cannot be used in this MIB as one of the values of
1381         jmAttributeValueAsInteger."
1382     SYNTAX      INTEGER(0..2147483647)
1383
1384
1385
1386
1387 JmJobSourcePlatformTypeTC ::= TEXTUAL-CONVENTION
1388     STATUS      current
1389     DESCRIPTION
1390         "The source platform type that can submit jobs to servers or
1391         devices in any of the 3 configurations."
1392     REFERENCE
1393         "This is a type 2 enumeration.  See Section 3.7.1.2.  See also
1394         IANA operating-system-names registry."
1395     SYNTAX      INTEGER {
1396         other(1),
1397         unknown(2),
1398         sptUNIX(3),           -- UNIX
1399         sptOS2(4),           -- OS/2
1400         sptPCDOS(5),         -- DOS
1401         sptNT(6),           -- NT
1402         sptMVS(7),          -- MVS
1403         sptVM(8),           -- VM
1404         sptOS400(9),        -- OS/400
1405         sptVMS(10),         -- VMS
1406         sptWindows(11),     -- Windows
1407         sptNetWare(12)      -- NetWare
1408     }

```

```
1398
1399 JmFinishingTC ::= TEXTUAL-CONVENTION
1400     STATUS          current
1401     DESCRIPTION
1402         "The type of finishing operation.
1403
1404         These values are the same as the enum values of the IPP
1405         'finishings' attribute.  See Section 3.7.1.2.
1406
1407         other(1),
1408             Some other finishing operation besides one of the specified
1409             or registered values.
1410
1411         unknown(2),
1412             The finishing is unknown.
1413
1414         none(3),
1415             Perform no finishing.
1416
1417         staple(4),
1418             Bind the document(s) with one or more staples. The exact
1419             number and placement of the staples is site-defined.
1420
1421         punch(5),
1422             This value indicates that holes are required in the
1423             finished document. The exact number and placement of the
1424             holes is site-defined. The punch specification MAY be
1425             satisfied (in a site- and implementation-specific manner)
1426             either by drilling/punching, or by substituting pre-drilled
1427             media.
1428
1429         cover(6),
1430             This value is specified when it is desired to select a non-
1431             printed (or pre-printed) cover for the document. This does
1432             not supplant the specification of a printed cover (on cover
1433             stock medium) by the document itself.
1434
1435         bind(7)
1436             This value indicates that a binding is to be applied to the
1437             document; the type and placement of the binding is product-
1438             specific."
1439     REFERENCE
1440         "This is a type 2 enumeration.  See Section 3.7.1.2."
1441     SYNTAX          INTEGER {
1442         other(1),
1443         unknown(2),
1444         none(3),
1445         staple(4),
1446         punch(5),
1447         cover(6),
1448         bind(7)
1449     }
```

```
1450
1451
1452 JmPrintQualityTC ::= TEXTUAL-CONVENTION
1453     STATUS          current
1454     DESCRIPTION
1455         "Print quality settings.
1456
1457         These values are the same as the enum values of the IPP 'print-
1458         quality' attribute.  See Section 3.7.1.2."
1459     REFERENCE
1460         "This is a type 2 enumeration.  See Section 3.7.1.2."
1461     SYNTAX          INTEGER {
1462         other(1),      -- Not one of the specified or registered
1463                        -- values.
1464         unknown(2),   -- The actual value is unknown.
1465         draft(3),     -- Lowest quality available on the printer.
1466         normal(4),    -- Normal or intermediate quality on the
1467                        -- printer.
1468         high(5)       -- Highest quality available on the printer.
1469     }
1470
1471
1472 JmPrinterResolutionTC ::= TEXTUAL-CONVENTION
1473     STATUS          current
1474     DESCRIPTION
1475         "Printer resolutions.
1476
1477         Nine octets consisting of two 4-octet SIGNED-INTEGERS followed
1478         by a SIGNED-BYTE.  The values are the same as those specified
1479         in the Printer MIB [printmib].  The first SIGNED-INTEGER
1480         contains the value of prtMarkerAddressabilityXFeedDir.  The
1481         second SIGNED-INTEGER contains the value of
1482         prtMarkerAddressabilityFeedDir.  The SIGNED-BYTE contains the
1483         value of prtMarkerAddressabilityUnit.
1484
1485         Note: the latter value is either 3 (tenThousandsOfInches) or 4
1486         (micrometers) and the addressability is in 10,000 units of
1487         measure.  Thus the SIGNED-INTEGERS represent integral values in
1488         either dots-per-inch or dots-per-centimeter.
1489
1490         The syntax is the same as the IPP 'printer-resolution'
1491         attribute.  See Section 3.7.1.2."
1492     SYNTAX          OCTET STRING (SIZE(9))
1493
```



```
1489
1490 JmTonerEconomyTC ::= TEXTUAL-CONVENTION
1491     STATUS      current
1492     DESCRIPTION
1493         "Toner economy settings."
1494     REFERENCE
1495         "This is a type 2 enumeration.  See Section 3.7.1.2."
1496     SYNTAX      INTEGER {
1497         unknown(2),      -- unknown.
1498         off(3),          -- Off. Normal. Use full toner.
1499         on(4)            -- On. Use less toner than normal.
1500     }
1501 JmBooleanTC ::= TEXTUAL-CONVENTION
1502     STATUS      current
1503     DESCRIPTION
1504         "Boolean true or false value."
1505     REFERENCE
1506         "This is a type 2 enumeration.  See Section 3.7.1.2."
1507     SYNTAX      INTEGER {
1508         unknown(2),      -- unknown.
1509         false(3),        -- FALSE.
1510         true(4)         -- TRUE.
1511     }
1512 JmMediumTypeTC ::= TEXTUAL-CONVENTION
1513     STATUS      current
1514     DESCRIPTION
1515         "Identifies the type of medium.
1516
1517         other(1),
1518             The type is neither one of the values listed in this
1519             specification nor a registered value.
1520
1521         unknown(2),
1522             The type is not known.
1523
1524         stationery(3),
1525             Separately cut sheets of an opaque material.
1526
1527         transparency(4),
1528             Separately cut sheets of a transparent material.
1529
1530         envelope(5),
1531             Envelopes that can be used for conventional mailing
1532             purposes.
```

```
1533
1534     envelopePlain(6),
1535         Envelopes that are not preprinted and have no windows.
1536
1537     envelopeWindow(7),
1538         Envelopes that have windows for addressing purposes.
1539
1540     continuousLong(8),
1541         Continuously connected sheets of an opaque material
1542         connected along the long edge.
1543
1544     continuousShort(9),
1545         Continuously connected sheets of an opaque material
1546         connected along the short edge.
1547
1548     tabStock(10),
1549         Media with tabs.
1550
1551     multiPartForm(11),
1552         Form medium composed of multiple layers not pre-attached to
1553         one another; each sheet MAY be drawn separately from an
1554         input source.
1555
1556     labels(12),
1557         Label-stock.
1558
1559     multiLayer(13)
1560         Form medium composed of multiple layers which are pre-
1561         attached to one another, e.g. for use with impact
1562         printers."
1563 REFERENCE
1564     "This is a type 2 enumeration. See Section 3.7.1.2. These
1565     enum values correspond to the keyword name strings of the
1566     prtInputMediaType object in the Printer MIB [print-mib]. There
1567     is no printer description attribute in IPP/1.0 that represents
1568     these values."
1569 SYNTAX     INTEGER {
1570     other(1),
1571     unknown(2),
1572     stationery(3),
1573     transparency(4),
1574     envelope(5),
1575     envelopePlain(6),
1576     envelopeWindow(7),
1577     continuousLong(8),
1578     continuousShort(9),
1579     tabStock(10),
1580     multiPartForm(11),
1581     labels(12),
1582     multiLayer(13)
1583 }
1584
```

```

1585
1586 JmJobCollationTypeTC ::= TEXTUAL-CONVENTION
1587     STATUS          current
1588     DESCRIPTION
1589         "This value is the type of job collation.  Implementations that
1590         don't support multiple documents or don't support multiple
1591         copies SHALL NOT support the uncollatedDocuments(5) value."
1592     REFERENCE
1593         "This is a type 2 enumeration.  See Section 3.7.1.2.  See also
1594         Section 3.4, entitled 'Monitoring Job Progress'."
1595     SYNTAX          INTEGER {
1596         other(1),
1597         unknown(2),
1598         uncollatedSheets(3),      -- sheets within each document copy
1599                                   -- are not collated: 1 1 ..., 2 2 ...,
1600         collatedDocuments(4),    -- internal collated sheets,
1601                                   -- documents: A, B, A, B, ...
1602         uncollatedDocuments(5)  -- internal collated sheets,
1603                                   -- documents: A, A, ..., B, B, ...
1604     }
1605
1606 JmJobSubmissionIDTypeTC ::= TEXTUAL-CONVENTION
1607     STATUS          current
1608     DESCRIPTION
1609         "Identifies the format type of a job submission ID.
1610
1611         Each job submission ID is a fixed-length, 48-octet printable
1612         US-ASCII [US-ASCII] coded character string containing no
1613         control characters, consisting of the following fields:
1614
1615         octet 1:  The format letter identifying the format.  The US-
1616         ASCII characters '0-9', 'A-Z', and 'a-z' are assigned in
1617         order giving 62 possible formats.
1618         octets 2-40:  A 39-character, US-ASCII trailing SPACE filled
1619         field specified by the format letter, if the data is less
1620         than 39 ASCII characters.
1621         octets 41-48:  A sequential or random US-ASCII number to make
1622         the ID quasi-unique.
1623
1624         If the client does not supply a job submission ID in the job
1625         submission protocol, then the agent SHALL assign a job
1626         submission ID using any of the standard formats that are
1627         reserved for the agent.  Clients SHALL not use formats that are
1628         reserved for agents and agents SHALL NOT use formats that are
1629         reserved for clients, in order to reduce conflicts in ID
1630         generation.  See the description for which formats are reserved
1631         for clients or for agents.
1632
1633         Registration of additional formats may be done following the
1634         procedures described in Section 3.7.3.
1635

```

1636 The format values defined at the time of completion of this
1637 specification are:
1638
1639 Format
1640 Letter Description
1641 -----
1642 '0' Job Owner generated by the server/device
1643 octets 2-40: The last 39 bytes of the jmJobOwner object.
1644 octets 41-48: The US-ASCII 8-decimal-digit sequential number
1645 assigned by the agent.
1646 This format is reserved for agents.
1647
1648 NOTE - Clients wishing to use a job submission ID that
1649 incorporates the job owner, SHALL use format '8', not
1650 format '0'.
1651
1652 '1' Job Name
1653 octets 2-40: The last 39 bytes of the jobName attribute.
1654 octets 41-48: The US-ASCII 8-decimal-digit random number
1655 assigned by the client.
1656 This format is reserved for clients.
1657
1658 '2' Client MAC address
1659 octets 2-40: The client MAC address: in hexadecimal with each
1660 nibble of the 6 octet address being '0'-'9' or 'A' - 'F'
1661 (uppercase only). Most significant octet first.
1662 octets 41-48: The US-ASCII 8-decimal-digit sequential number
1663 assigned by the client.
1664 This format is reserved for clients.
1665
1666 '3' Client URL
1667 octets 2-40: The last 39 bytes of the client URL [URI-spec].
1668 octets 41-48: The US-ASCII 8-decimal-digit sequential number
1669 assigned by the client.
1670 This format is reserved for clients.
1671
1672 '4' Job URI
1673 octets 2-40: The last 39 bytes of the URI [URI-spec] assigned
1674 by the server or device to the job when the job was
1675 submitted for processing.
1676 octets 41-48: The US-ASCII 8-decimal-digit sequential number
1677 assigned by the agent.
1678 This format is reserved for agents.
1679
1680 '5' POSIX User Number
1681 octets 2-40: The last 39 bytes of a user number, such as POSIX
1682 user number.
1683 octets 41-48: The US-ASCII 8-decimal-digit sequential number
1684 assigned by the client.
1685 This format is reserved for clients.
1686

1687 '6' User Account Number
1688 octets 2-40: The last 39 bytes of the user account number.
1689 octets 41-48: The US-ASCII 8-decimal-digit sequential number
1690 assigned by the client.
1691 This format is reserved for clients.
1692
1693 '7' DTMF Incoming FAX routing number
1694 octets 2-40: The last 39 bytes of the DTMF incoming FAX
1695 routing number.
1696 octets 41-48: The US-ASCII 8-decimal-digit sequential number
1697 assigned by the client.
1698 This format is reserved for clients.
1699
1700 '8' Job Owner supplied by the client
1701 octets 2-40: The last 39 bytes of the job owner name (that the
1702 agent returns in the jmJobOwner object).
1703 octets 41-48: The US-ASCII 8-decimal-digit sequential number
1704 assigned by the client.
1705 This format is reserved for clients. See format '0' which is
1706 reserved for agents.
1707
1708 '9' Host Name
1709 octets 2-40: The last 39 bytes of the host name with trailing
1710 SPACES that submitted the job to this server/device using a
1711 protocol, such as LPD [RFC-1179] which includes the host
1712 name in the job submission protocol.
1713 octets 41-48: The US-ASCII 8-decimal-digit leading zero
1714 representation of the job id generated by the submitting
1715 server (configuration 3) or the client (configuration 1 and
1716 2), such as in the LPD protocol.
1717 This format is reserved for clients.
1718
1719 'A' AppleTalk Protocol
1720 octets 2-40: Contains the AppleTalk printer name, with the
1721 first character of the name in octet 2. AppleTalk printer
1722 names are a maximum of 31 characters. Any unused portion
1723 of this field shall be filled with spaces.
1724 octets 41-48: '00000XXX', where 'XXX' is the 3-digit US-ASCII
1725 decimal representation of the Connection Id.
1726 This format is reserved for agents.
1727
1728 'B' NetWare PServer
1729 octets 2-40: Contains the Directory Path Name as recorded by
1730 the Novell File Server in the queue directory. If the
1731 string is less than 40 octets, the left-most character in
1732 the string shall appear in octet position 2. Otherwise,
1733 only the last 39 bytes shall be included. Any unused
1734 portion of this field shall be filled with spaces.
1735 octets 41-48: '000XXXXX' The US-ASCII representation of the
1736 Job Number as per the NetWare File Server Queue Management
1737 Services.
1738 This format is reserved for agents.

1739
1740 'C' Server Message Block protocol (SMB)
1741 octets 2-40: Contains a decimal (US-ASCII coded)
1742 representation of the 16 bit SMB Tree Id field, which
1743 uniquely identifies the connection that submitted the job
1744 to the printer. The most significant digit of the numeric
1745 string shall be placed in octet position 2. All unused
1746 portions of this field shall be filled with spaces. The
1747 SMB Tree Id has a maximum value of 65,535.
1748 octets 41-48: The US-ASCII 8-decimal-digit leading zero
1749 representation of the File Handle returned from the device
1750 to the client in response to a Create Print File command.
1751 This format is reserved for agents.
1752
1753 'D' Transport Independent Printer/System Interface (TIP/SI)
1754 octets 2-40: Contains the Job Name from the Job Control-Start
1755 Job (JC-SJ) command. If the Job Name portion is less than
1756 40 octets, the left-most character in the string shall
1757 appear in octet position 2. Any unused portion of this
1758 field shall be filled with spaces. Otherwise, only the
1759 last 39 bytes shall be included.
1760 octets 41-48: The US-ASCII 8-decimal-digit leading zero
1761 representation of the jmJobIndex assigned by the agent.
1762 This format is reserved for agents, since the agent supplies
1763 octets 41-48, though the client supplies the job name. See
1764 format '1' reserved to clients to submit job name ids in
1765 which they supply octets 41-48.
1766
1767 NOTE - the job submission id is only intended to be unique
1768 between a limited set of clients for a limited duration of
1769 time, namely, for the life time of the job in the context of
1770 the server or device that is processing the job. Some of the
1771 formats include something that is unique per client and a
1772 random number so that the same job submitted by the same client
1773 will have a different job submission id. For other formats,
1774 where part of the id is guaranteed to be unique for each
1775 client, such as the MAC address or URL, a sequential number
1776 SHOULD suffice for each client (and may be easier for each
1777 client to manage). Therefore, the length of the job submission
1778 id has been selected to reduce the probability of collision to
1779 an extremely low number, but is not intended to be an absolute
1780 guarantee of uniqueness. None-the-less, collisions are
1781 remotely possible, but without bad consequences, since this MIB
1782 is intended to be used only for monitoring jobs, not for
1783 controlling and managing them."
1784 REFERENCE
1785 "This is like a type 2 enumeration. See section 3.7.3."
1786 SYNTAX OCTET STRING(SIZE(1)) -- ASCII '0'-'9', 'A'-'Z', 'a'-'z'

```

1787
1788 JmJobStateTC ::= TEXTUAL-CONVENTION
1789     STATUS      current
1790     DESCRIPTION
1791         "The current state of the job (pending, processing, completed,
1792         etc.).
1793
1794         The following figure shows the normal job state transitions:
1795
1796                                     +----> canceled(7)
1797                                     /
1798     +----> pending(3) -----> processing(5) -----+-----> completed(9)
1799     |           ^           |           ^           |
1800     |           |           |           |           |
1801     |           v           |           v           |
1802     +----> pendingHeld(4)  processingStopped(6) ----+
1803

```

1804 Figure 4 - Normal Job State Transitions

1805
 1806 Normally a job progresses from left to right. Other state
 1807 transitions are unlikely, but are not forbidden. Not shown are
 1808 the transitions to the canceled state from the pending,
 1809 pendingHeld, and processingStopped states.

1810
 1811 Jobs in the pending, processing, and processingStopped states
 1812 are called 'active', while jobs in the pendingHeld, canceled,
 1813 aborted, and completed states are called 'inactive'. Jobs
 1814 reach one of the three terminal states: completed, canceled, or
 1815 aborted, *after* the jobs have completed all activity, and all
 1816 MIB objects and attributes have reached their final values for
 1817 the job.

1818
 1819 These values are the same as the enum values of the IPP 'job-
 1820 state' job attribute. See Section 3.7.1.2.

1821
 1822 unknown(2),
 1823 The job state is *not* known, or its state is indeterminate.

1824
 1825 pending(3),
 1826 The job is a candidate to start processing, but is not yet
 1827 processing.

1828
 1829 pendingHeld(4),
 1830 The job is not a candidate for processing for any number of
 1831 reasons but will return to the pending state as soon as the
 1832 reasons are no longer present. The job's
 1833 jmJobStateReasons1 object and/or jobStateReasonsN (N=2..4)
 1834 attributes SHALL indicate why the job is no longer a
 1835 candidate for processing. The reasons are represented as
 1836 bits in the jmJobStateReasons1 object and/or
 1837 jobStateReasonsN (N=2..4) attributes. See the

1838 JmJobStateReasonsMTC (N=1..4) textual convention for the
1839 specification of each reason.

1840
1841 processing(5),

1842 One or more of:

1843
1844 1. the job is using, or is attempting to use, one or more
1845 purely software processes that are analyzing, creating, or
1846 interpreting a PDL, etc.,

1847
1848 2. the job is using, or is attempting to use, one or more
1849 hardware devices that are interpreting a PDL, making marks
1850 on a medium, and/or performing finishing, such as stapling,
1851 etc.,

1852
1853 OR

1854
1855 3. (configuration 2) the server has made the job ready for
1856 printing, but the output device is not yet printing it,
1857 either because the job hasn't reached the output device or
1858 because the job is queued in the output device or some
1859 other spooler, awaiting the output device to print it.

1860
1861 When the job is in the processing state, the entire job
1862 state includes the detailed status represented in the
1863 device MIB indicated by the hrDeviceIndex value of the
1864 job's physicalDevice attribute, if the agent implements
1865 such a device MIB.

1866
1867 Implementations MAY, though they NEED NOT, include
1868 additional values in the job's jmJobStateReasons1 object to
1869 indicate the progress of the job, such as adding the
1870 jobPrinting value to indicate when the device is actually
1871 making marks on a medium and/or the processingToStopPoint
1872 value to indicate that the server or device is in the
1873 process of canceling or aborting the job.

1874
1875 processingStopped(6),

1876 The job has stopped while processing for any number of
1877 reasons and will return to the processing state as soon as
1878 the reasons are no longer present.

1879
1880 The job's jmJobStateReasons1 object and/or the job's
1881 jobStateReasonsN (N=2..4) attributes MAY indicate why the
1882 job has stopped processing. For example, if the output
1883 device is stopped, the deviceStopped value MAY be included
1884 in the job's jmJobStateReasons1 object.

1885
1886 NOTE - When an output device is stopped, the device usually
1887 indicates its condition in human readable form at the
1888 device. The management application can obtain more
1889 complete device status remotely by querying the appropriate

1890 device MIB using the job's deviceIndex attribute(s), if the
 1891 agent implements such a device MIB

1892
 1893 canceled(7),

1894 A client has canceled the job and the server or device has
 1895 completed canceling the job AND all MIB objects and
 1896 attributes have reached their final values for the job.
 1897 While the server or device is canceling the job, the job's
 1898 jmJobStateReasons1 object SHOULD contain the
 1899 processingToStopPoint value and one of the canceledByUser,
 1900 canceledByOperator, or canceledAtDevice values. The
 1901 canceledByUser, canceledByOperator, or canceledAtDevice
 1902 values remain while the job is in the canceled state.
 1903

1904 aborted(8),

1905 The job has been aborted by the system, usually while the
 1906 job was in the processing or processingStopped state and
 1907 the server or device has completed aborting the job AND all
 1908 MIB objects and attributes have reached their final values
 1909 for the job. While the server or device is aborting the
 1910 job, the job's jmJobStateReasons1 object MAY contain the
 1911 processingToStopPoint and abortedBySystem values. If
 1912 implemented, the abortedBySystem value SHALL remain while
 1913 the job is in the aborted state.
 1914

1915 completed(9)

1916 The job has completed successfully or with warnings or
 1917 errors after processing and all of the media have been
 1918 successfully stacked in the appropriate output bin(s) AND
 1919 all MIB objects and attributes have reached their final
 1920 values for the job. The job's jmJobStateReasons1 object
 1921 SHOULD contain one of: completedSuccessfully,
 1922 completedWithWarnings, or completedWithErrors values."
 1923

1924 REFERENCE

1925 "This is a type 2 enumeration. See Section 3.7.1.2."

1926 SYNTAX INTEGER {
 1927 unknown(2),
 1928 pending(3),
 1929 pendingHeld(4),
 1930 processing(5),
 1931 processingStopped(6),
 1932 canceled(7),
 1933 aborted(8),
 1934 completed(9)
 }

1935
 1936 JmAttributeTypeTC ::= TEXTUAL-CONVENTION
 1937 STATUS current
 1938 DESCRIPTION
 1939 "The type of the attribute which identifies the attribute.
 1940
 1941 In the following definitions of the enums, each description
 1942 indicates whether the useful value of the attribute SHALL be
 1943 represented using the jmAttributeValueAsInteger or the
 1944 jmAttributeValueAsOctets objects by the initial tag: 'INTEGER:'
 1945 or 'OCTETS:', respectively.
 1946
 1947 Some attributes allow the agent implementer a choice of useful
 1948 values of either an integer, an octets representation, or both,
 1949 depending on implementation. These attributes are indicated
 1950 with 'INTEGER:' AND/OR 'OCTETS:' tags.
 1951
 1952 A very few attributes require both objects at the same time to
 1953 represent a pair of useful values (see mediumConsumed(171)).
 1954 These attributes are indicated with 'INTEGER:' AND 'OCTETS:'
 1955 tags. See the jmAttributeGroup for the descriptions of these
 1956 two MANDATORY objects.
 1957
 1958 NOTE - The enum assignments are grouped logically with values
 1959 assigned in groups of 20, so that additional values may be
 1960 registered in the future and assigned a value that is part of
 1961 their logical grouping.
 1962
 1963 Values in the range 2**30 to 2**31-1 are reserved for private
 1964 or experimental usage. This range corresponds to the same
 1965 range reserved in IPP. Implementers are warned that use of
 1966 such values may conflict with other implementations.
 1967 Implementers are encouraged to request registration of enum
 1968 values following the procedures in Section 3.7.1.
 1969
 1970 NOTE: No attribute name exceeds 31 characters.
 1971
 1972 The standard attribute types defined at the time of completion
 1973 of the specification are:

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

1981
 1982
 1983

1984 ++++++

1985 + Job State attributes

1986 +

1987 + The following attributes specify the state of a job.

1988 ++++++

1989

1990 jobStateReasons2(3), JmJobStateReasons2TC

1991 INTEGER: Additional information about the job's current

1992 state that augments the jmJobState object. See the

1993 description under the JmJobStateReasons1TC textual-

1994 convention.

1995

1996 jobStateReasons3(4), JmJobStateReasons3TC

1997 INTEGER: Additional information about the job's current

1998 state that augments the jmJobState object. See the

1999 description under JmJobStateReasons1TC textual-convention.

2000

2001 jobStateReasons4(5), JmJobStateReasons4TC

2002 INTEGER: Additional information about the job's current

2003 state that augments the jmJobState object. See the

2004 description under JmJobStateReasons1TC textual-convention.

2005

2006 processingMessage(6), JmUTF8StringTC(SIZE(0..63))

2007 OCTETS: MULTI-ROW: A coded character set message that is

2008 generated by the server or device during the processing of

2009 the job as a simple form of processing log to show progress

2010 and any problems. The natural language of each value is

2011 specified by the corresponding

2012 processingMessageNaturalLangTag(7) value.

2013

2014 NOTE - This attribute is intended for such conditions as

2015 interpreter messages, rather than being the printable form

2016 of the jmJobState and jmJobStateReasons1 objects and

2017 jobStateReasons2, jobStateReasons3, and jobStateReasons4

2018 attributes. In order to produce a localized printable form

2019 of these job state objects/attribute, a management

2020 application SHOULD produce a message from their enum and

2021 bit values.

2022

2023 NOTE - There is no job description attribute in IPP/1.0

2024 that corresponds to this attribute and this attribute does

2025 not correspond to the IPP/1.0 'job-state-message' job

2026 description attribute, which is just a printable form of

2027 the IPP 'job-state' and 'job-state-reasons' job attributes.

2028

2029 There is no restriction for the same message occurring in

2030 multiple rows.

2031

2032 processingMessageNaturalLangTag(7), OCTET STRING(SIZE(0..63))
2033 OCTETS: MULTI-ROW: The natural language of the
2034 corresponding processingMessage(6) attribute value. See
2035 section 3.6.1, entitled 'Text generated by the server or
2036 device'.
2037

2038 If the agent does not know the natural language of the job
2039 processing message, the agent SHALL either (1) return a
2040 zero length string value for the
2041 processingMessageNaturalLangTag(7) attribute or (2) not
2042 return the processingMessageNaturalLangTag(7) attribute for
2043 the job.
2044

2045 There is no restriction for the same tag occurring in
2046 multiple rows, since when this attribute is implemented, it
2047 SHOULD have a value row for each corresponding
2048 processingMessage(6) attribute value row.
2049

2050 jobCodedCharSet(8), CodedCharSet
2051 INTEGER: The MIBenum identifier of the coded character set
2052 that the agent is using to represent coded character set
2053 objects and attributes of type 'JmJobStringTC'. These
2054 coded character set objects and attributes are either: (1)
2055 supplied by the job submitting client or (2) defaulted by
2056 the server or device when omitted by the job submitting
2057 client. The agent SHALL represent these objects and
2058 attributes in the MIB either (1) in the coded character set
2059 as they were submitted or (2) MAY convert the coded
2060 character set to another coded character set or encoding
2061 scheme as identified by the jobCodedCharSet(8) attribute.
2062 See section 3.6.2, entitled 'Text supplied by the job
2063 submitter'.
2064

2065 These MIBenum values are assigned by IANA [IANA-charsets]
2066 when the coded character sets are registered. The coded
2067 character set SHALL be one of the ones registered with IANA
2068 [IANA] and the enum value uses the CodedCharSet textual-
2069 convention from the Printer MIB. See the JmJobStringTC
2070 textual-convention.
2071

2072 If the agent does not know what coded character set was
2073 used by the job submitting client, the agent SHALL either
2074 (1) return the 'unknown(2)' value for the
2075 jobCodedCharSet(8) attribute or (2) not return the
2076 jobCodedCharSet(8) attribute for the job.
2077

2078 jobNaturalLanguageTag(9), OCTET STRING(SIZE(0..63))
2079 OCTETS: The natural language of the job attributes supplied
2080 by the job submitter or defaulted by the server or device
2081 for the job, i.e., all objects and attributes represented
2082 by the 'JmJobStringTC' textual-convention, such as jobName,
2083 mediumRequested, etc. See Section 3.6.2, entitled 'Text
2084 supplied by the job submitter'.
2085

2086 If the agent does not know what natural language was used
2087 by the job submitting client, the agent SHALL either (1)
2088 return a zero length string value for the
2089 jobNaturalLanguageTag(9) attribute or (2) not return
2090 jobNaturalLanguageTag(9) attribute for the job.
2091

2092
2093 +++++
2094 + Job Identification attributes
2095 +
2096 + The following attributes help an end user, a system
2097 + operator, or an accounting program identify a job.
2098 +++++

2099
2100
2101
2102 jobURI(20), OCTET STRING(SIZE(0..63))
2103 OCTETS: MULTI-ROW: The job's Universal Resource
2104 Identifier (URI) [RFC-1738]. See IPP [ipp-model] for
2105 example usage.
2106

2107 NOTE - The agent may be able to generate this value on each
2108 SNMP Get operation from smaller values, rather than having
2109 to store the entire URI.
2110

2111 If the URI exceeds 63 octets, the agent SHALL use multiple
2112 values, with the next 63 octets coming in the second value,
2113 etc.
2114

2115 NOTE - IPP [ipp-model] has a 1023-octet maximum length for
2116 a URI, though the URI standard itself and HTTP/1.1 specify
2117 no maximum length.
2118

2119 jobAccountName(21), OCTET STRING(SIZE(0..63))
2120 OCTETS: Arbitrary binary information which MAY be coded
2121 character set data or encrypted data supplied by the
2122 submitting user for use by accounting services to allocate
2123 or categorize charges for services provided, such as a
2124 customer account name or number.
2125

2126 NOTE: This attribute NEED NOT be printable characters.
2127

2128 serverAssignedJobName(22), JmJobStringTC(SIZE(0..63))
2129 OCTETS: Configuration 3 only: The human readable string
2130 name, number, or ID of the job as assigned by the server
2131 that submitted the job to the device that the agent is
2132 providing access to with this MIB.
2133
2134 NOTE - This attribute is intended for enabling a user to
2135 find his/her job that a server submitted to a device when
2136 either the client does not support the jmJobSubmissionID or
2137 the server does not pass the jmJobSubmissionID through to
2138 the device.
2139

2140 jobName(23), JmJobStringTC(SIZE(0..63))
2141 OCTETS: The human readable string name of the job as
2142 assigned by the submitting user to help the user
2143 distinguish between his/her various jobs. This name does
2144 not need to be unique.
2145
2146 This attribute is intended for enabling a user or the
2147 user's application to convey a job name that MAY be printed
2148 on a start sheet, returned in a query result, or used in
2149 notification or logging messages.
2150
2151 In order to assist users to find their jobs for job
2152 submission protocols that don't supply a jmJobSubmissionID,
2153 the agent SHOULD maintain the jobName attribute for the
2154 time specified by the jmGeneralJobPersistence object,
2155 rather than the (shorter) jmGeneralAttributePersistence
2156 object.
2157
2158 If this attribute is not specified when the job is
2159 submitted, no job name is assumed, but implementation
2160 specific defaults are allowed, such as the value of the
2161 documentName attribute of the first document in the job or
2162 the fileName attribute of the first document in the job.
2163
2164 The jobName attribute is distinguished from the jobComment
2165 attribute, in that the jobName attribute is intended to
2166 permit the submitting user to distinguish between different
2167 jobs that he/she has submitted. The jobComment attribute
2168 is intended to be free form additional information that a
2169 user might wish to use to communicate with himself/herself,
2170 such as a reminder of what to do with the results or to
2171 indicate a different set of input parameters were tried in
2172 several different job submissions.
2173

2174 jobServiceTypes(24), JmJobServiceTypesTC
2175 INTEGER: Specifies the type(s) of service to which the job
2176 has been submitted (print, fax, scan, etc.). The service
2177 type is bit encoded with each job service type so that more
2178 general and arbitrary services can be created, such as
2179 services with more than one destination type, or ones with
2180 only a source or only a destination. For example, a job
2181 service might scan, faxOut, and print a single job. In
2182 this case, three bits would be set in the jobServiceTypes
2183 attribute, corresponding to the hexadecimal values: 0x8 +
2184 0x20 + 0x4, respectively, yielding: 0x2C.
2185
2186 Whether this attribute is set from a job attribute supplied
2187 by the job submission client or is set by the recipient job
2188 submission server or device depends on the job submission
2189 protocol. This attribute SHALL be implemented if the
2190 server or device has other types in addition to or instead
2191 of printing.
2192
2193 One of the purposes of this attribute is to permit a
2194 requester to filter out jobs that are not of interest. For
2195 example, a printer operator may only be interested in jobs
2196 that include printing.
2197
2198 jobSourceChannelIndex(25), Integer32(0..2147483647)
2199 INTEGER: The index of the row in the associated Printer
2200 MIB[print-mib] of the channel which is the source of the
2201 print job.
2202
2203 jobSourcePlatformType(26), JmJobSourcePlatformTypeTC
2204 INTEGER: The source platform type of the immediate
2205 upstream submitter that submitted the job to the server
2206 (configuration 2) or device (configuration 1 and 3) to
2207 which the agent is providing access. For configuration 1,
2208 this is the type of the client that submitted the job to
2209 the device; for configuration 2, this is the type of the
2210 client that submitted the job to the server; and for
2211 configuration 3, this is the type of the server that
2212 submitted the job to the device.
2213
2214 submittingServerName(27), JmJobStringTC(SIZE(0..63))
2215 OCTETS: For configuration 3 only: The administrative name
2216 of the server that submitted the job to the device.
2217
2218 submittingApplicationName(28), JmJobStringTC(SIZE(0..63))
2219 OCTETS: The name of the client application (not the server
2220 in configuration 3) that submitted the job to the server or
2221 device.
2222

2223 jobOriginatingHost(29), JmJobStringTC(SIZE(0..63))
2224 OCTETS: The name of the client host (not the server host
2225 name in configuration 3) that submitted the job to the
2226 server or device.
2227

2228 deviceNameRequested(30), JmJobStringTC(SIZE(0..63))
2229 OCTETS: The administratively defined coded character set
2230 name of the target device requested by the submitting user.
2231 For configuration 1, its value corresponds to the Printer
2232 MIB[print-mib]: prtGeneralPrinterName object. For
2233 configuration 2 and 3, its value is the name of the logical
2234 or physical device that the user supplied to indicate to
2235 the server on which device(s) they wanted the job to be
2236 processed.
2237

2238 queueNameRequested(31), JmJobStringTC(SIZE(0..63))
2239 OCTETS: The administratively defined coded character set
2240 name of the target queue requested by the submitting user.
2241 For configuration 1, its value corresponds to the queue in
2242 the device for which the agent is providing access. For
2243 configuration 2 and 3, its value is the name of the queue
2244 that the user supplied to indicate to the server on which
2245 device(s) they wanted the job to be processed.
2246

2247 NOTE - typically an implementation SHOULD support either
2248 the deviceNameRequested or queueNameRequested attribute,
2249 but not both.
2250

2251 physicalDevice(32), hrDeviceIndex
2252 AND/OR
2253 JmUTF8StringTC(SIZE(0..63))
2254 INTEGER: MULTI-ROW: The index of the physical device MIB
2255 instance requested/used, such as the Printer MIB[print-
2256 mib]. This value is an hrDeviceIndex value. See the Host
2257 Resources MIB[hr-mib].
2258

2259 AND/OR
2260

2261 OCTETS: MULTI-ROW: The name of the physical device to
2262 which the job is assigned.
2263

2264 numberOfDocuments(33), Integer32(-2..2147483647)
2265 INTEGER: The number of documents in this job.
2266

2267 The agent SHOULD return this attribute if the job has more
2268 than one document.
2269

2270 fileName(34), JmJobStringTC(SIZE(0..63))
2271 OCTETS: MULTI-ROW: The coded character set file name or
2272 URI[URI-spec] of the document.
2273
2274 There is no restriction on the same file name occurring in
2275 multiple rows.
2276
2277 documentName(35), JmJobStringTC(SIZE(0..63))
2278 OCTETS: MULTI-ROW: The coded character set name of the
2279 document.
2280
2281 There is no restriction on the same document name occurring
2282 in multiple rows.
2283
2284 jobComment(36), JmJobStringTC(SIZE(0..63))
2285 OCTETS: An arbitrary human-readable coded character text
2286 string supplied by the submitting user or the job
2287 submitting application program for any purpose. For
2288 example, a user might indicate what he/she is going to do
2289 with the printed output or the job submitting application
2290 program might indicate how the document was produced.
2291
2292 The jobComment attribute is not intended to be a name; see
2293 the jobName attribute.
2294
2295 documentFormatIndex(37), Integer32(0..2147483647)
2296 INTEGER: MULTI-ROW: The index in the prtInterpreterTable
2297 in the Printer MIB[print-mib] of the page description
2298 language (PDL) or control language interpreter that this
2299 job requires/uses. A document or a job MAY use more than
2300 one PDL or control language.
2301
2302 NOTE - As with all intensive attributes where multiple rows
2303 are allowed, there SHALL be only one distinct row for each
2304 distinct interpreter; there SHALL be no duplicates.
2305
2306 NOTE - This attribute type is intended to be used with an
2307 agent that implements the Printer MIB and SHALL not be used
2308 if the agent does not implement the Printer MIB. Such an
2309 agent SHALL use the documentFormat attribute instead.
2310

```

2311     documentFormat(38),                               PrtInterpreterLangFamilyTC
2312                                                     AND/OR
2313                                                     OCTET STRING(SIZE(0..63))
2314     INTEGER: MULTI-ROW: The interpreter language family
2315     corresponding to the Printer MIB[print-mib]
2316     prtInterpreterLangFamily object, that this job
2317     requires/uses. A document or a job MAY use more than one
2318     PDL or control language.
2319
2320     AND/OR
2321
2322     OCTETS: MULTI-ROW: The document format registered as a
2323     media type[iana-media-types], i.e., the name of the MIME
2324     content-type/subtype. Examples: 'application/postscript',
2325     'application/vnd.hp-PCL', 'application/pdf', 'text/plain'
2326     (US-ASCII SHALL be assumed), 'text/plain; charset=iso-8859-
2327     1', and 'application/octet-stream'. The IPP 'document-
2328     format' job attribute uses these same values with the same
2329     semantics. See the IPP [ipp-model] 'mimeMediaType'
2330     attribute syntax and the document-format attribute for
2331     further examples and explanation.
2332
2333
2334     ++++++
2335     + Job Parameter attributes
2336     +
2337     + The following attributes represent input parameters
2338     + supplied by the submitting client in the job submission
2339     + protocol.
2340     ++++++
2341
2342     jobPriority(50),                                     Integer32(-2..100)
2343     INTEGER: The priority for scheduling the job. It is used
2344     by servers and devices that employ a priority-based
2345     scheduling algorithm.
2346
2347     A higher value specifies a higher priority. The value 1 is
2348     defined to indicate the lowest possible priority (a job
2349     which a priority-based scheduling algorithm SHALL pass over
2350     in favor of higher priority jobs). The value 100 is
2351     defined to indicate the highest possible priority.
2352     Priority is expected to be evenly or 'normally' distributed
2353     across this range. The mapping of vendor-defined priority
2354     over this range is implementation-specific. -2 indicates
2355     unknown.
2356

```

2357 jobProcessAfterDateAndTime(51), DateAndTime (SNMPv2-TC)
2358 OCTETS: The calendar date and time of day after which the
2359 job SHALL become a candidate to be scheduled for
2360 processing. If the value of this attribute is in the
2361 future, the server SHALL set the value of the job's
2362 jmJobState object to pendingHeld and add the
2363 jobProcessAfterSpecified bit value to the job's
2364 jmJobStateReasons1 object. When the specified date and
2365 time arrives, the server SHALL remove the
2366 jobProcessAfterSpecified bit value from the job's
2367 jmJobStateReasons1 object and, if no other reasons remain,
2368 SHALL change the job's jmJobState object to pending.
2369
2370 jobHold(52), JmBooleanTC
2371 INTEGER: If the value is 'true(4)', a client has
2372 explicitly specified that the job is to be held until
2373 explicitly released. Until the job is explicitly released
2374 by a client, the job SHALL be in the pendingHeld state with
2375 the jobHoldSpecified value in the jmJobStateReasons1
2376 attribute.
2377
2378 jobHoldUntil(53), JmJobStringTC(SIZE(0..63))
2379 OCTETS: The named time period during which the job SHALL
2380 become a candidate for processing, such as 'evening',
2381 'night', 'weekend', 'second-shift', 'third-shift', etc., as
2382 defined by the system administrator. See IPP [ipp-model]
2383 for the standard keyword values. Until that time period
2384 arrives, the job SHALL be in the pendingHeld state with the
2385 jobHoldUntilSpecified value in the jmJobStateReasons1
2386 object. The value 'no-hold' SHALL indicate explicitly that
2387 no time period has been specified; the absence of this
2388 attribute SHALL indicate implicitly that no time period has
2389 been specified.
2390
2391 outputBin(54), Integer32(0..2147483647)
2392 AND/OR
2393 JmJobStringTC(SIZE(0..63))
2394 INTEGER: MULTI-ROW: The output subunit index in the
2395 Printer MIB[print-mib]
2396
2397 AND/OR
2398
2399 OCTETS: MULTI-ROW: the name or number (represented as
2400 ASCII digits) of the output bin to which all or part of the
2401 job is placed in.
2402

```

2403 sides(55), Integer32(-2..2)
2404     INTEGER: MULTI-ROW: The number of sides, '1' or '2', that
2405     any document in this job requires/used.
2406
2407 finishing(56), JmFinishingTC
2408     INTEGER: MULTI-ROW: Type of finishing that any document
2409     in this job requires/used.
2410
2411
2412 ++++++
2413 + Image Quality attributes (requested and consumed)
2414 +
2415 + For devices that can vary the image quality.
2416 ++++++
2417
2418 printQualityRequested(70), JmPrintQualityTC
2419     INTEGER: MULTI-ROW: The print quality selection requested
2420     for a document in the job for printers that allow quality
2421     differentiation.
2422
2423 printQualityUsed(71), JmPrintQualityTC
2424     INTEGER: MULTI-ROW: The print quality selection actually
2425     used by a document in the job for printers that allow
2426     quality differentiation.
2427
2428 printerResolutionRequested(72), JmPrinterResolutionTC
2429     OCTETS: MULTI-ROW: The printer resolution requested for a
2430     document in the job for printers that support resolution
2431     selection.
2432
2433 printerResolutionUsed(73), JmPrinterResolutionTC
2434     OCTETS: MULTI-ROW: The printer resolution actually used
2435     by a document in the job for printers that support
2436     resolution selection.
2437
2438 tonerEcomonyRequested(74), JmTonerEcomonyTC
2439     INTEGER: MULTI-ROW: The toner economy selection requested
2440     for documents in the job for printers that allow toner
2441     economy differentiation.
2442
2443 tonerEcomonyUsed(75), JmTonerEcomonyTC
2444     INTEGER: MULTI-ROW: The toner economy selection actually
2445     used by documents in the job for printers that allow toner
2446     economy differentiation.
2447
2448 tonerDensityRequested(76), Integer32(-2..100)
2449     INTEGER: MULTI-ROW: The toner density requested for a
2450     document in this job for devices that can vary toner
2451     density levels. Level 1 is the lowest density and level
2452     100 is the highest density level. Devices with a smaller
2453     range, SHALL map the 1-100 range evenly onto the
2454     implemented range.

```

2455
2456 tonerDensityUsed(77), Integer32(-2..100)
2457 INTEGER: MULTI-ROW: The toner density used by documents
2458 in this job for devices that can vary toner density levels.
2459 Level 1 is the lowest density and level 100 is the highest
2460 density level. Devices with a smaller range, SHALL map the
2461 1-100 range evenly onto the implemented range.
2462
2463
2464 ++++++
2465 + Job Progress attributes (requested and consumed)
2466 +
2467 + Pairs of these attributes can be used by monitoring
2468 + applications to show an indication of relative progress
2469 + to users. See section 3.4, entitled
2470 + 'Monitoring Job Progress'.
2471 ++++++
2472
2473 jobCopiesRequested(90), Integer32(-2..2147483647)
2474 INTEGER: The number of copies of the entire job that are
2475 to be produced.
2476
2477 jobCopiesCompleted(91), Integer32(-2..2147483647)
2478 INTEGER: The number of copies of the entire job that have
2479 been completed so far.
2480
2481 documentCopiesRequested(92), Integer32(-2..2147483647)
2482 INTEGER: The total count of the number of document copies
2483 requested for the job as a whole. If there are documents
2484 A, B, and C, and document B is specified to produce 4
2485 copies, the number of document copies requested is 6 for
2486 the job.
2487
2488 This attribute SHALL be used only when a job has multiple
2489 documents. The jobCopiesRequested attribute SHALL be used
2490 when the job has only one document.
2491
2492 documentCopiesCompleted(93), Integer32(-2..2147483647)
2493 INTEGER: The total count of the number of document copies
2494 completed so far for the job as a whole. If there are
2495 documents A, B, and C, and document B is specified to
2496 produce 4 copies, the number of document copies starts a 0
2497 and runs up to 6 for the job as the job processes.
2498
2499 This attribute SHALL be used only when a job has multiple
2500 documents. The jobCopiesCompleted attribute SHALL be used
2501 when the job has only one document.
2502

2503 jobKOctetsTransferred(94), Integer32(-2..2147483647)
2504 INTEGER: The number of K (1024) octets transferred to the
2505 server or device to which the agent is providing access.
2506 This count is independent of the number of copies of the
2507 job or documents that will be produced, but it is only a
2508 measure of the number of bytes transferred to the server or
2509 device.
2510
2511 The agent SHALL round the actual number of octets
2512 transferred up to the next higher K. Thus 0 octets SHALL
2513 be represented as '0', 1-1024 octets SHALL BE represented
2514 as '1', 1025-2048 SHALL be '2', etc. When the job
2515 completes, the values of the jmJobKOctetsPerCopyRequested
2516 object and the jobKOctetsTransferred attribute SHALL be
2517 equal.
2518
2519 NOTE - The jobKOctetsTransferred can be used with the
2520 jmJobKOctetsPerCopyRequested object in order to produce a
2521 relative indication of the progress of the job for agents
2522 that do not implement the jmJobKOctetsProcessed object.
2523
2524 sheetCompletedCopyNumber(95), Integer32(-2..2147483647)
2525 INTEGER: The number of the copy being stacked for the
2526 current document. This number starts at 0, is set to 1
2527 when the first sheet of the first copy for each document is
2528 being stacked and is equal to n where n is the nth sheet
2529 stacked in the current document copy. See section 3.4 ,
2530 entitled 'Monitoring Job Progress'.
2531
2532 sheetCompletedDocumentNumber(96), Integer32(-2..2147483647)
2533 INTEGER: The ordinal number of the document in the job
2534 that is currently being stacked. This number starts at 0,
2535 increments to 1 when the first sheet of the first document
2536 in the job is being stacked, and is equal to n where n is
2537 the nth document in the job, starting with 1.
2538
2539 Implementations that only support one document jobs SHOULD
2540 NOT implement this attribute.
2541
2542 jobCollationType(97), JmJobCollationTypeTC
2543 INTEGER: The type of job collation. See also Section 3.4,
2544 entitled 'Monitoring Job Progress'.
2545

```
2546
2547      ++++++
2548      + Impression attributes
2549      +
2550      + See the definition of the terms 'impression', 'sheet',
2551      + and 'page' in Section 2.
2552      +
2553      + See also jmJobImpressionsPerCopyRequested and
2554      + jmJobImpressionsCompleted objects in the jmJobTable.
2555      ++++++
2556
2557      impressionsSpooled(110),          Integer32(-2..2147483647)
2558          INTEGER: The number of impressions spooled to the server
2559          or device for the job so far.
2560
2561      impressionsSentToDevice(111),     Integer32(-2..2147483647)
2562          INTEGER: The number of impressions sent to the device for
2563          the job so far.
2564
2565      impressionsInterpreted(112),      Integer32(-2..2147483647)
2566          INTEGER: The number of impressions interpreted for the job
2567          so far.
2568
2569      impressionsCompletedCurrentCopy(113), Integer32(-2..2147483647)
2570          INTEGER: The number of impressions completed by the device
2571          for the current copy of the current document so far. For
2572          printing, the impressions completed includes interpreting,
2573          marking, and stacking the output. For other types of job
2574          services, the number of impressions completed includes the
2575          number of impressions processed.
2576
2577          This value SHALL be reset to 0 for each document in the job
2578          and for each document copy.
2579
2580      fullColorImpressionsCompleted(114), Integer32(-2..2147483647)
2581          INTEGER: The number of full color impressions completed by
2582          the device for this job so far. For printing, the
2583          impressions completed includes interpreting, marking, and
2584          stacking the output. For other types of job services, the
2585          number of impressions completed includes the number of
2586          impressions processed. Full color impressions are typically
2587          defined as those requiring 3 or more colorants, but this
2588          MAY vary by implementation. In any case, the value of this
2589          attribute counts by 1 for each side that has full color,
2590          not by the number of colors per side (and the other
2591          impression counters are incremented, except
2592          highlightColorImpressionsCompleted(115)).
2593
```

2594 highlightColorImpressionsCompleted(115),
 2595 Integer32(-2..2147483647)
 2596 INTEGER: The number of highlight color impressions
 2597 completed by the device for this job so far. For printing,
 2598 the impressions completed includes interpreting, marking,
 2599 and stacking the output. For other types of job services,
 2600 the number of impressions completed includes the number of
 2601 impressions processed. Highlight color impressions are
 2602 typically defined as those requiring black plus one other
 2603 colorant, but this MAY vary by implementation. In any
 2604 case, the value of this attribute counts by 1 for each side
 2605 that has highlight color (and the other impression counters
 2606 are incremented, except
 2607 fullColorImpressionsCompleted(114)).
 2608
 2609
 2610 ++++++
 2611 + Page attributes
 2612 +
 2613 + See the definition of 'impression', 'sheet', and 'page'
 2614 + in Section 2.
 2615 ++++++
 2616
 2617 pagesRequested(130), Integer32(-2..2147483647)
 2618 INTEGER: The number of logical pages requested by the job
 2619 to be processed.
 2620
 2621 pagesCompleted(131), Integer32(-2..2147483647)
 2622 INTEGER: The number of logical pages completed for this
 2623 job so far.
 2624
 2625 For implementations where multiple copies are produced by
 2626 the interpreter with only a single pass over the data, the
 2627 final value SHALL be equal to the value of the
 2628 pagesRequested object. For implementations where multiple
 2629 copies are produced by the interpreter by processing the
 2630 data for each copy, the final value SHALL be a multiple of
 2631 the value of the pagesRequested object.
 2632
 2633 NOTE - See the impressionsCompletedCurrentCopy and
 2634 pagesCompletedCurrentCopy attributes for attributes that
 2635 are reset on each document copy.
 2636
 2637 NOTE - The pagesCompleted object can be used with the
 2638 pagesRequested object to provide an indication of the
 2639 relative progress of the job, provided that the
 2640 multiplicative factor is taken into account for some
 2641 implementations of multiple copies.
 2642

2643 pagesCompletedCurrentCopy(132), Integer32(-2..2147483647)
2644 INTEGER: The number of logical pages completed for the
2645 current copy of the document so far. This value SHALL be
2646 reset to 0 for each document in the job and for each
2647 document copy.
2648
2649
2650 ++++++
2651 + Sheet attributes
2652 +
2653 + See the definition of 'impression', 'sheet', and 'page'
2654 + in Section 2.
2655 ++++++

2656
2657 sheetsRequested(150), Integer32(-2..2147483647)
2658 INTEGER: The total number of medium sheets requested to be
2659 produced for this job.
2660
2661 Unlike the jmJobKOctetsPerCopyRequested and
2662 jmJobImpressionsPerCopyRequested attributes, the
2663 sheetsRequested(150) attribute SHALL include the
2664 multiplicative factor contributed by the number of copies
2665 and so is the total number of sheets to be produced by the
2666 job, as opposed to the size of the document(s) submitted.
2667

2668 sheetsCompleted(151), Integer32(-2..2147483647)
2669 INTEGER: The total number of medium sheets that have
2670 completed marking and stacking for the entire job so far
2671 whether those sheets have been processed on one side or on
2672 both.
2673

2674 sheetsCompletedCurrentCopy(152), Integer32(-2..2147483647)
2675 INTEGER: The number of medium sheets that have completed
2676 marking and stacking for the current copy of a document in
2677 the job so far whether those sheets have been processed on
2678 one side or on both.
2679
2680 The value of this attribute SHALL be 0 before the job
2681 starts processing and SHALL be reset to 1 after the first
2682 sheet of each document and document copy in the job is
2683 processed and stacked.
2684
2685

```

2686      ++++++
2687      + Resources attributes (requested and consumed)
2688      +
2689      + Pairs of these attributes can be used by monitoring
2690      + applications to show an indication of relative usage to
2691      + users.
2692      ++++++
2693
2694      mediumRequested(170),                JmMediumTypeTC
2695                                          AND/OR
2696                                          JmJobStringTC(SIZE(0..63))
2697      INTEGER:  MULTI-ROW:  The type
2698      AND/OR
2699      OCTETS:   MULTI-ROW:  the name of the medium that is
2700      required by the job.
2701
2702      NOTE - The name (JmJobStringTC) values correspond to the
2703      prtInputMediaName object in the Printer MIB [print-mib] and
2704      the values of the IPP 'media' attribute.
2705
2706      mediumConsumed(171),                Integer32(-2..2147483647)
2707                                          AND
2708                                          JmJobStringTC(SIZE(0..63))
2709      INTEGER:  MULTI-ROW:  The number of sheets
2710      AND
2711      OCTETS:   MULTI-ROW:  the name of the medium that has been
2712      consumed so far whether those sheets have been processed on
2713      one side or on both.
2714
2715      This attribute SHALL have both Integer32 and OCTET STRING
2716      (represented as JmJobStringTC) values.
2717
2718      NOTE - The name (JmJobStringTC) values correspond to the
2719      name values of the prtInputMediaName object in the Printer
2720      MIB [print-mib].
2721
2722      colorantRequested(172),             Integer32(-2..2147483647)
2723                                          AND/OR
2724                                          JmJobStringTC(SIZE(0..63))
2725      INTEGER:  MULTI-ROW:  The index (prtMarkerColorantIndex) in
2726      the Printer MIB[print-mib]
2727      AND/OR
2728      OCTETS:   MULTI-ROW:  the name of the colorant requested.
2729
2730      NOTE - The name (JmJobStringTC) values correspond to the
2731      name values of the prtMarkerColorantValue object in the
2732      Printer MIB.  Examples are: red, blue.

```

```

2733         colorantConsumed(173),           Integer32(-2..2147483647)
2734                                         AND/OR
2735                                         JmJobStringTC(SIZE(0..63))
2736         INTEGER:  MULTI-ROW:  The index (prtMarkerColorantIndex) in
2737         the Printer MIB[print-mib]
2738         AND/OR
2739         OCTETS:  MULTI-ROW:  the name of the colorant consumed.
2740
2741         NOTE - The name (JmJobStringTC) values correspond to the
2742         name values of the prtMarkerColorantValue object in the
2743         Printer MIB.  Examples are: red, blue
2744
2745
2746         ++++++
2747         + Time attributes (set by server or device)
2748         +
2749         + This section of attributes are ones that are set by the
2750         + server or device that accepts jobs.  Two forms of time are
2751         + provided.  Each form is represented in a separate attribute.
2752         + See section 3.1.2 and section 3.1.3 for the
2753         + conformance requirements for time attribute for agents and
2754         + monitoring applications, respectively.  The two forms are:
2755         +
2756         + 'DateAndTime' is an 8 or 11 octet binary encoded year,
2757         + month, day, hour, minute, second, deci-second with
2758         + optional offset from UTC.  See SNMPv2-TC [SMIV2-TC].
2759         +
2760         + NOTE: 'DateAndTime' is not printable characters; it is
2761         + binary.
2762         +
2763         + 'JmTimeStampTC' is the time of day measured in the number of
2764         + seconds since the system was booted.
2765         ++++++
2766
2767         jobSubmissionToServerTime(190),     JmTimeStampTC
2768                                         AND/OR
2769                                         DateAndTime
2770         INTEGER:  Configuration 3 only:  The time
2771         AND/OR
2772         OCTETS:  the date and time that the job was submitted to
2773         the server (as distinguished from the device which uses
2774         jobSubmissionTime).
2775
2776         jobSubmissionTime(191),           JmTimeStampTC
2777                                         AND/OR
2778                                         DateAndTime
2779         INTEGER:  Configurations 1, 2, and 3:  The time
2780         AND/OR
2781         OCTETS:  the date and time that the job was submitted to
2782         the server or device to which the agent is providing
2783         access.
2784

```

```

2785     jobStartedBeingHeldTime(192),      JmTimeStampTC
2786                                         AND/OR
2787                                         DateAndTime
2788     INTEGER:  The time
2789     AND/OR
2790     OCTETS:  the date and time that the job last entered the
2791     pendingHeld state.  If the job has never entered the
2792     pendingHeld state, then the value SHALL be '0' or the
2793     attribute SHALL not be present in the table.
2794
2795     jobStartedProcessingTime(193),      JmTimeStampTC
2796                                         AND/OR
2797                                         DateAndTime
2798     INTEGER:  The time
2799     AND/OR
2800     OCTETS:  the date and time that the job started processing.
2801
2802     jobCompletionTime(194),             JmTimeStampTC
2803                                         AND/OR
2804                                         DateAndTime
2805     INTEGER:  The time
2806     AND/OR
2807     OCTETS:  the date and time that the job entered the
2808     completed, canceled, or aborted state.
2809
2810     jobProcessingCPUTime(195)           Integer32(-2..2147483647)
2811     UNITS    'seconds'
2812     INTEGER:  The amount of CPU time in seconds that the job
2813     has been in the processing state.  If the job enters the
2814     processingStopped state, that elapsed time SHALL not be
2815     included.  In other words, the jobProcessingCPUTime value
2816     SHOULD be relatively repeatable when the same job is
2817     processed again on the same device."
2818
2819     REFERENCE
2820     "See Section 3.2 entitled 'The Attribute Mechanism' for a
2821     description of this textual-convention and its use in the
2822     jmAttributeTable.
2823
2824     This is a type 2 enumeration.  See Section 3.7.1.2."
2825     SYNTAX    INTEGER {
2826     other(1),
2827
2828     -- Job State attributes:
2829     jobStateReasons2(3),
2830     jobStateReasons3(4),
2831     jobStateReasons4(5),
2832     processingMessage(6),
2833     processingMessageNaturalLangTag(7),
2834     jobCodedCharSet(8),
2835     jobNaturalLanguageTag(9),
2836

```

```
2837     -- Job Identification attributes:
2838     jobURI(20),
2839     jobAccountName(21),
2840     serverAssignedJobName(22),
2841     jobName(23),
2842     jobServiceTypes(24),
2843     jobSourceChannelIndex(25),
2844     jobSourcePlatformType(26),
2845     submittingServerName(27),
2846     submittingApplicationName(28),
2847     jobOriginatingHost(29),
2848     deviceNameRequested(30),
2849     queueNameRequested(31),
2850     physicalDevice(32),
2851     numberOfDocuments(33),
2852     fileName(34),
2853     documentName(35),
2854     jobComment(36),
2855     documentFormatIndex(37),
2856     documentFormat(38),
2857
2858     -- Job Parameter attributes:
2859     jobPriority(50),
2860     jobProcessAfterDateAndTime(51),
2861     jobHold(52),
2862     jobHoldUntil(53),
2863     outputBin(54),
2864     sides(55),
2865     finishing(56),
2866
2867     -- Image Quality attributes:
2868     printQualityRequested(70),
2869     printQualityUsed(71),
2870     printerResolutionRequested(72),
2871     printerResolutionUsed(73),
2872     tonerEcomonyRequested(74),
2873     tonerEcomonyUsed(75),
2874     tonerDensityRequested(76),
2875     tonerDensityUsed(77),
2876
2877     -- Job Progress attributes:
2878     jobCopiesRequested(90),
2879     jobCopiesCompleted(91),
2880     documentCopiesRequested(92),
2881     documentCopiesCompleted(93),
2882     jobKOctetsTransferred(94),
2883     sheetCompletedCopyNumber(95),
2884     sheetCompletedDocumentNumber(96),
2885     jobCollationType(97),
2886
```

```
2887     -- Impression attributes:
2888     impressionsSpooled(110),
2889     impressionsSentToDevice(111),
2890     impressionsInterpreted(112),
2891     impressionsCompletedCurrentCopy(113),
2892     fullColorImpressionsCompleted(114),
2893     highlightColorImpressionsCompleted(115),
2894
2895     -- Page attributes:
2896     pagesRequested(130),
2897     pagesCompleted(131),
2898     pagesCompletedCurrentCopy(132),
2899
2900     -- Sheet attributes:
2901     sheetsRequested(150),
2902     sheetsCompleted(151),
2903     sheetsCompletedCurrentCopy(152),
2904
2905     -- Resource attributes:
2906     mediumRequested(170),
2907     mediumConsumed(171),
2908     colorantRequested(172),
2909     colorantConsumed(173),
2910
2911     -- Time attributes:
2912     jobSubmissionToServerTime(190),
2913     jobSubmissionTime(191),
2914     jobStartedBeingHeldTime(192),
2915     jobStartedProcessingTime(193),
2916     jobCompletionTime(194),
2917     jobProcessingCPUtime(195)
2918 }
2919
2920
2921
2922
```

```

2923 JmJobServiceTypesTC ::= TEXTUAL-CONVENTION
2924     STATUS      current
2925     DESCRIPTION
2926         "Specifies the type(s) of service to which the job has been
2927         submitted (print, fax, scan, etc.). The service type is
2928         represented as an enum that is bit encoded with each job
2929         service type so that more general and arbitrary services can be
2930         created, such as services with more than one destination type,
2931         or ones with only a source or only a destination. For example,
2932         a job service might scan, faxOut, and print a single job. In
2933         this case, three bits would be set in the jobServiceTypes
2934         attribute, corresponding to the hexadecimal values: 0x8 + 0x20
2935         + 0x4, respectively, yielding: 0x2C.
2936
2937         Whether this attribute is set from a job attribute supplied by
2938         the job submission client or is set by the recipient job
2939         submission server or device depends on the job submission
2940         protocol. With either implementation, the agent SHALL return a
2941         non-zero value for this attribute indicating the type of the
2942         job.
2943
2944         One of the purposes of this attribute is to permit a requester
2945         to filter out jobs that are not of interest. For example, a
2946         printer operator MAY only be interested in jobs that include
2947         printing. That is why the attribute is in the job
2948         identification category.
2949
2950         The following service component types are defined (in
2951         hexadecimal) and are assigned a separate bit value for use with
2952         the jobServiceTypes attribute:
2953
2954         other                0x1
2955             The job contains some instructions that are not one of the
2956             identified types.
2957
2958         unknown             0x2
2959             The job contains some instructions whose type is unknown to
2960             the agent.
2961
2962         print                0x4
2963             The job contains some instructions that specify printing
2964
2965         scan                 0x8
2966             The job contains some instructions that specify scanning
2967
2968         faxIn                0x10
2969             The job contains some instructions that specify receive fax
2970
2971         faxOut              0x20
2972             The job contains some instructions that specify sending fax
2973

```

2974 getFile 0x40
2975 The job contains some instructions that specify accessing
2976 files or documents
2977
2978 putFile 0x80
2979 The job contains some instructions that specify storing
2980 files or documents
2981
2982 mailList 0x100
2983 The job contains some instructions that specify
2984 distribution of documents using an electronic mail system."
2985 REFERENCE
2986 "These bit definitions are the equivalent of a type 2 enum
2987 except that combinations of them MAY be used together. See
2988 section 3.7.1.2."
2989 SYNTAX INTEGER(0..2147483647) -- 31 bits, all but sign bit
2990
2991
2992
2993 JmJobStateReasons1TC ::= TEXTUAL-CONVENTION
2994 STATUS current
2995 DESCRIPTION
2996 "The JmJobStateReasonsMTC (*N*=1..4) textual-conventions are used
2997 with the jmJobStateReasons1 object and jobStateReasonsN
2998 (*N*=2..4), respectively, to provide additional information
2999 regarding the current jmJobState object value. These values
3000 MAY be used with any job state or states for which the reason
3001 makes sense.
3002
3003 NOTE - While values cannot be added to the jmJobState object
3004 without impacting deployed clients that take actions upon
3005 receiving jmJobState values, it is the intent that additional
3006 JmJobStateReasonsMTC enums can be defined and registered
3007 without impacting such deployed clients. In other words, the
3008 jmJobStateReasons1 object and jobStateReasonsN attributes are
3009 intended to be extensible.
3010
3011 NOTE - The Job Monitoring MIB contains a superset of the IPP
3012 values[ipp-model] for the IPP 'job-state-reasons' attribute,
3013 since the Job Monitoring MIB is intended to cover other job
3014 submission protocols as well. Also some of the names of the
3015 reasons have been changed from 'printer' to 'device', since the
3016 Job Monitoring MIB is intended to cover additional types of
3017 devices, including input devices, such as scanners.
3018
3019 The following standard values are defined (in hexadecimal) as
3020 *powers of two*, since multiple values MAY be used at the same
3021 time. For ease of understanding, the JmJobStateReasons1TC
3022 reasons are presented in the order in which the reasons are
3023 likely to occur (if implemented), starting with the
3024 'jobIncoming' value and ending with the
3025 'jobCompletedWithErrors' value.

3026
3027 other 0x1
3028 The job state reason is not one of the standardized or
3029 registered reasons.
3030
3031 unknown 0x2
3032 The job state reason is not known to the agent or is
3033 indeterminent.
3034
3035 jobIncoming 0x4
3036 The job has been accepted by the server or device, but the
3037 server or device is expecting (1) additional operations
3038 from the client to finish creating the job and/or (2) is
3039 accessing/accepting document data.
3040
3041 submissionInterrupted 0x8
3042 The job was not completely submitted for some unforeseen
3043 reason, such as: (1) the server has crashed before the job
3044 was closed by the client, (2) the server or the document
3045 transfer method has crashed in some non-recoverable way
3046 before the document data was entirely transferred to the
3047 server, (3) the client crashed or failed to close the job
3048 before the time-out period.
3049
3050 jobOutgoing 0x10
3051 Configuration 2 only: The server is transmitting the job
3052 to the device.
3053
3054 jobHoldSpecified 0x20
3055 The value of the job's jobHold(52) attribute is TRUE. The
3056 job SHALL NOT be a candidate for processing until this
3057 reason is removed and there are no other reasons to hold
3058 the job.
3059
3060 jobHoldUntilSpecified 0x40
3061 The value of the job's jobHoldUntil(53) attribute specifies
3062 a time period that is still in the future. The job SHALL
3063 NOT be a candidate for processing until this reason is
3064 removed and there are no other reasons to hold the job.
3065
3066 jobProcessAfterSpecified 0x80
3067 The value of the job's jobProcessAfterDateAndTime(51)
3068 attribute specifies a time that is still in the future.
3069 The job SHALL NOT be a candidate for processing until this
3070 reason is removed and there are no other reasons to hold
3071 the job.
3072

3073 resourcesAreNotReady 0x100
3074 At least one of the resources needed by the job, such as
3075 media, fonts, resource objects, etc., is not ready on any
3076 of the physical devices for which the job is a candidate.
3077 This condition MAY be detected when the job is accepted, or
3078 subsequently while the job is pending or processing,
3079 depending on implementation.
3080

3081 deviceStoppedPartly 0x200
3082 One or more, but not all, of the devices to which the job
3083 is assigned are stopped. If all of the devices are stopped
3084 (or the only device is stopped), the deviceStopped reason
3085 SHALL be used.
3086

3087 deviceStopped 0x400
3088 The device(s) to which the job is assigned is (are all)
3089 stopped.
3090

3091 jobInterpreting 0x800
3092 The device to which the job is assigned is interpreting the
3093 document data.
3094

3095 jobPrinting 0x1000
3096 The output device to which the job is assigned is marking
3097 media. This attribute is useful for servers and output
3098 devices which spend a great deal of time processing (1)
3099 when no marking is happening and then want to show that
3100 marking is now happening or (2) when the job is in the
3101 process of being canceled or aborted while the job remains
3102 in the processing state, but the marking has not yet
3103 stopped so that impression or sheet counts are still
3104 increasing for the job.
3105

3106 jobCanceledByUser 0x2000
3107 The job was canceled by the owner of the job, i.e., by a
3108 user whose name is the same as the value of the job's
3109 jmJobOwner object, or by some other authorized end-user,
3110 such as a member of the job owner's security group.
3111

3112 jobCanceledByOperator 0x4000
3113 The job was canceled by the operator, i.e., by a user who
3114 has been authenticated as having operator privileges
3115 (whether local or remote).
3116

3117 jobCanceledAtDevice 0x8000
3118 The job was canceled by an unidentified local user, i.e., a
3119 user at a console at the device.
3120

3121 abortedBySystem 0x10000
3122 The job (1) is in the process of being aborted, (2) has
3123 been aborted by the system and placed in the 'aborted'
3124 state, or (3) has been aborted by the system and placed in
3125 the 'pendingHeld' state, so that a user or operator can
3126 manually try the job again.
3127

3128 processingToStopPoint 0x20000
3129 The requester has issued an operation to cancel or
3130 interrupt the job or the server/device has aborted the job,
3131 but the server/device is still performing some actions on
3132 the job until a specified stop point occurs or job
3133 termination/cleanup is completed.
3134
3135 This reason is recommended to be used in conjunction with
3136 the processing job state to indicate that the server/device
3137 is still performing some actions on the job while the job
3138 remains in the processing state. After all the job's
3139 resources consumed counters have stopped incrementing, the
3140 server/device moves the job from the processing state to
3141 the canceled or aborted job states.
3142

3143 serviceOffLine 0x40000
3144 The service or document transform is off-line and accepting
3145 no jobs. All pending jobs are put into the pendingHeld
3146 state. This situation could be true if the service's or
3147 document transform's input is impaired or broken.
3148

3149 jobCompletedSuccessfully 0x80000
3150 The job completed successfully.
3151

3152 jobCompletedWithWarnings 0x100000
3153 The job completed with warnings.
3154

3155 jobCompletedWithErrors 0x200000
3156 The job completed with errors (and possibly warnings too).
3157
3158

3159 The following additional job state reasons have been added to
3160 represent job states that are in ISO DPA[iso-dpa] and other job
3161 submission protocols:
3162

3163 jobPaused 0x400000
3164 The job has been indefinitely suspended by a client issuing
3165 an operation to suspend the job so that other jobs may
3166 proceed using the same devices. The client MAY issue an
3167 operation to resume the paused job at any time, in which
3168 case the agent SHALL remove the jobPaused values from the
3169 job's jmJobStateReasons1 object and the job is eventually
3170 resumed at or near the point where the job was paused.
3171

```

3172     jobInterrupted                0x800000
3173     The job has been interrupted while processing by a client
3174     issuing an operation that specifies another job to be run
3175     instead of the current job.  The server or device will
3176     automatically resume the interrupted job when the
3177     interrupting job completes.
3178
3179     jobRetained                      0x1000000
3180     The job is being retained by the server or device with all
3181     of the job's document data (and submitted resources, such
3182     as fonts, logos, and forms, if any).  Thus a client could
3183     issue an operation to the server or device to either (1)
3184     re-do the job (or a copy of the job) on the same server or
3185     device or (2) resubmit the job to another server or device.
3186     When a client could no longer re-do/resubmit the job, such
3187     as after the document data has been discarded, the agent
3188     SHALL remove the jobRetained value from the
3189     jmJobStateReasons1 object."
3190 REFERENCE
3191     "These bit definitions are the equivalent of a type 2 enum
3192     except that combinations of bits may be used together.  See
3193     section 3.7.1.2.  The remaining bits are reserved for future
3194     standardization and/or registration."
3195 SYNTAX     INTEGER(0..2147483647)    -- 31 bits, all but sign bit
3196
3197
3198
3199 JmJobStateReasons2TC ::= TEXTUAL-CONVENTION
3200     STATUS      current
3201     DESCRIPTION
3202     "This textual-convention is used with the jobStateReasons2
3203     attribute to provides additional information regarding the
3204     jmJobState object.  See the description under
3205     JmJobStateReasons1TC for additional information that applies to
3206     all reasons.
3207
3208     The following standard values are defined (in hexadecimal) as
3209     powers of two, since multiple values may be used at the same
3210     time:
3211
3212     cascaded                0x1
3213     An outbound gateway has transmitted all of the job's job
3214     and document attributes and data to another spooling
3215     system.
3216
3217     deletedByAdministrator  0x2
3218     The administrator has deleted the job.
3219
3220     discardTimeArrived      0x4
3221     The job has been deleted due to the fact that the time
3222     specified by the job's job-discard-time attribute has
3223     arrived.

```

3224
3225 postProcessingFailed 0x8
3226 The post-processing agent failed while trying to log
3227 accounting attributes for the job; therefore the job has
3228 been placed into the completed state with the jobRetained
3229 jmJobStateReasons1 object value for a system-defined period
3230 of time, so the administrator can examine it, resubmit it,
3231 etc.
3232
3233 jobTransforming 0x10
3234 The server/device is interpreting document data and
3235 producing another electronic representation.
3236
3237 maxJobFaultCountExceeded 0x20
3238 The job has faulted several times and has exceeded the
3239 administratively defined fault count limit.
3240
3241 devicesNeedAttentionTimeOut 0x40
3242 One or more document transforms that the job is using needs
3243 human intervention in order for the job to make progress,
3244 but the human intervention did not occur within the site-
3245 settable time-out value.
3246
3247 needsKeyOperatorTimeOut 0x80
3248 One or more devices or document transforms that the job is
3249 using need a specially trained operator (who may need a key
3250 to unlock the device and gain access) in order for the job
3251 to make progress, but the key operator intervention did not
3252 occur within the site-settable time-out value.
3253
3254 jobStartWaitTimeOut 0x100
3255 The server/device has stopped the job at the beginning of
3256 processing to await human action, such as installing a
3257 special cartridge or special non-standard media, but the
3258 job was not resumed within the site-settable time-out value
3259 and the server/device has transitioned the job to the
3260 pendingHeld state.
3261
3262 jobEndWaitTimeOut 0x200
3263 The server/device has stopped the job at the end of
3264 processing to await human action, such as removing a
3265 special cartridge or restoring standard media, but the job
3266 was not resumed within the site-settable time-out value and
3267 the server/device has transitioned the job to the completed
3268 state.
3269
3270 jobPasswordWaitTimeOut 0x400
3271 The server/device has stopped the job at the beginning of
3272 processing to await input of the job's password, but the
3273 password was not received within the site-settable time-out
3274 value.
3275

3276 deviceTimedOut 0x800
3277 A device that the job was using has not responded in a
3278 period specified by the device's site-settable attribute.
3279

3280 connectingToDeviceTimeOut 0x1000
3281 The server is attempting to connect to one or more devices
3282 which may be dial-up, polled, or queued, and so may be busy
3283 with traffic from other systems, but server was unable to
3284 connect to the device within the site-settable time-out
3285 value.
3286

3287 transferring 0x2000
3288 The job is being transferred to a down stream server or
3289 downstream device.
3290

3291 queuedInDevice 0x4000
3292 The server/device has queued the job in a down stream
3293 server or downstream device.
3294

3295 jobQueued 0x8000
3296 The server/device has queued the document data.
3297

3298 jobCleanup 0x10000
3299 The server/device is performing cleanup activity as part of
3300 ending normal processing.
3301

3302 jobPasswordWait 0x20000
3303 The server/device has selected the job to be next to
3304 process, but instead of assigning resources and starting
3305 the job processing, the server/device has transitioned the
3306 job to the pendingHeld state to await entry of a password
3307 (and dispatched another job, if there is one).
3308

3309 validating 0x40000
3310 The server/device is validating the job *after* accepting the
3311 job.
3312

3313 queueHeld 0x80000
3314 The operator has held the entire job set or queue.
3315

3316 jobProofWait 0x100000
3317 The job has produced a single proof copy and is in the
3318 pendingHeld state waiting for the requester to issue an
3319 operation to release the job to print normally, obeying any
3320 job and document copy attributes that were originally
3321 submitted.
3322

3323 heldForDiagnostics 0x200000
3324 The system is running intrusive diagnostics, so that all
3325 jobs are being held.

```

3326     noSpaceOnServer                0x800000
3327         There is no room on the server to store all of the job.
3328
3329     pinRequired                      0x1000000
3330         The System Administrator settable device policy is (1) to
3331         require PINs, and (2) to hold jobs that do not have a pin
3332         supplied as an input parameter when the job was created.
3333
3334     exceededAccountLimit            0x2000000
3335         The account for which this job is drawn has exceeded its
3336         limit. This condition SHOULD be detected before the job is
3337         scheduled so that the user does not wait until his/her job
3338         is scheduled only to find that the account is overdrawn.
3339         This condition MAY also occur while the job is processing
3340         either as processing begins or part way through processing.
3341
3342     heldForRetry                    0x4000000
3343         The job encountered some errors that the server/device
3344         could not recover from with its normal retry procedures,
3345         but the error might not be encountered if the job is
3346         processed again in the future. Example cases are phone
3347         number busy or remote file system in-accessible. For such
3348         a situation, the server/device SHALL transition the job
3349         from the processing to the pendingHeld, rather than to the
3350         aborted state.
3351
3352     The following values are from the X/Open PSIS draft standard:
3353
3354     canceledByShutdown              0x8000000
3355         The job was canceled because the server or device was
3356         shutdown before completing the job.
3357
3358     deviceUnavailable              0x10000000
3359         This job was aborted by the system because the device is
3360         currently unable to accept jobs.
3361
3362     wrongDevice                    0x20000000
3363         This job was aborted by the system because the device is
3364         unable to handle this particular job; the spooler SHOULD
3365         try another device or the user should submit the job to
3366         another device.
3367
3368     badJob                          0x40000000
3369         This job was aborted by the system because this job has a
3370         major problem, such as an ill-formed PDL; the spooler
3371         SHOULD not even try another device. "
3372     REFERENCE
3373         "These bit definitions are the equivalent of a type 2 enum
3374         except that combinations of them may be used together. See
3375         section 3.7.1.2. See the description under
3376         JmJobStateReasons1TC and the jobStateReasons2 attribute."
3377     SYNTAX      INTEGER(0..2147483647)  -- 31 bits, all but sign bit

```

3378
3379 JmJobStateReasons3TC ::= TEXTUAL-CONVENTION
3380 STATUS current
3381 DESCRIPTION
3382 "This textual-convention is used with the jobStateReasons3
3383 attribute to provides additional information regarding the
3384 jmJobState object. See the description under
3385 JmJobStateReasons1TC for additional information that applies to
3386 all reasons.
3387
3388 The following standard values are defined (in hexadecimal) as
3389 *powers of two*, since multiple values may be used at the same
3390 time:
3391
3392 jobInterruptedByDeviceFailure 0x1
3393 A device or the print system software that the job was
3394 using has failed while the job was processing. The server
3395 or device is keeping the job in the pendingHeld state until
3396 an operator can determine what to do with the job."
3397 REFERENCE
3398 "These bit definitions are the equivalent of a type 2 enum
3399 except that combinations of them may be used together. See
3400 section 3.7.1.2. The remaining bits are reserved for future
3401 standardization and/or registration. See the description under
3402 JmJobStateReasons1TC and the jobStateReasons3 attribute."
3403 SYNTAX INTEGER(0..2147483647) -- 31 bits, all but sign bit
3404
3405
3406
3407
3408
3409 JmJobStateReasons4TC ::= TEXTUAL-CONVENTION
3410 STATUS current
3411 DESCRIPTION
3412 "This textual-convention is used in the jobStateReasons4
3413 attribute to provides additional information regarding the
3414 jmJobState object. See the description under
3415 JmJobStateReasons1TC for additional information that applies to
3416 all reasons.
3417
3418 The following standard values are defined (in hexadecimal) as
3419 *powers of two*, since multiple values may be used at the same
3420 time:
3421
3422 none yet defined. These bits are reserved for future
3423 standardization and/or registration."
3424 REFERENCE
3425 "These bit definitions are the equivalent of a type 2 enum
3426 except that combinations of them may be used together. See
3427 section 3.7.1.2. See the description under
3428 JmJobStateReasons1TC and the jobStateReasons4 attribute."
3429 SYNTAX INTEGER(0..2147483647) -- 31 bits, all but sign bit


```

3430
3431 jobmonMIBObjects OBJECT IDENTIFIER ::= { jobmonMIB 1 }
3432
3433 -- The General Group (MANDATORY)
3434
3435 -- The jmGeneralGroup consists entirely of the jmGeneralTable.
3436
3437 jmGeneral OBJECT IDENTIFIER ::= { jobmonMIBObjects 1 }
3438
3439 jmGeneralTable OBJECT-TYPE
3440     SYNTAX      SEQUENCE OF JmGeneralEntry
3441     MAX-ACCESS  not-accessible
3442     STATUS      current
3443     DESCRIPTION
3444         "The jmGeneralTable consists of information of a general nature
3445         that are per-job-set, but are not per-job. See Section 2
3446         entitled 'Terminology and Job Model' for the definition of a
3447         job set."
3448     REFERENCE
3449         "The MANDATORY-GROUP macro specifies that this group is
3450         MANDATORY."
3451     ::= { jmGeneral 1 }
3452
3453
3454 jmGeneralEntry OBJECT-TYPE
3455     SYNTAX      JmGeneralEntry
3456     MAX-ACCESS  not-accessible
3457     STATUS      current
3458     DESCRIPTION
3459         "Information about a job set (queue).
3460
3461         An entry SHALL exist in this table for each job set."
3462     INDEX      { jmGeneralJobSetIndex }
3463     ::= { jmGeneralTable 1 }
3464
3465
3466 JmGeneralEntry ::= SEQUENCE {
3467     jmGeneralJobSetIndex      Integer32(1..32767),
3468     jmGeneralNumberOfActiveJobs Integer32(0..2147483647),
3469     jmGeneralOldestActiveJobIndex Integer32(0..2147483647),
3470     jmGeneralNewestActiveJobIndex Integer32(0..2147483647),
3471     jmGeneralJobPersistence    Integer32(15..2147483647),
3472     jmGeneralAttributePersistence Integer32(15..2147483647),
3473     jmGeneralJobSetName        JmUTF8StringTC(SIZE(0..63))
3474 }
3475

```

```
3476 jmGeneralJobSetIndex OBJECT-TYPE
3477     SYNTAX      Integer32(1..32767)
3478     MAX-ACCESS  not-accessible
3479     STATUS      current
3480     DESCRIPTION
3481         "A unique value for each job set in this MIB.  The jmJobTable
3482         and jmAttributeTable tables have this same index as their
3483         primary index.
3484
3485         The value(s) of the jmGeneralJobSetIndex SHALL be persistent
3486         across power cycles, so that clients that have retained
3487         jmGeneralJobSetIndex values will access the same job sets upon
3488         subsequent power-up.
3489
3490         An implementation that has only one job set, such as a printer
3491         with a single queue, SHALL hard code this object with the value
3492         1."
3493     REFERENCE
3494         "See Section 2 entitled 'Terminology and Job Model' for the
3495         definition of a job set.
3496         Corresponds to the first index in jmJobTable and
3497         jmAttributeTable."
3498     ::= { jmGeneralEntry 1 }
3499
3500
3501 jmGeneralNumberOfActiveJobs OBJECT-TYPE
3502     SYNTAX      Integer32(0..2147483647)
3503     MAX-ACCESS  read-only
3504     STATUS      current
3505     DESCRIPTION
3506         "The current number of 'active' jobs in the jmJobIDTable,
3507         jmJobTable, and jmAttributeTable, i.e., the total number of
3508         jobs that are in the pending, processing, or processingStopped
3509         states.  See the JmJobStateTC textual-convention for the exact
3510         specification of the semantics of the job states."
3511     DEFVAL     { 0 }      -- no jobs
3512     ::= { jmGeneralEntry 2 }
3513
```

```
3514 jmGeneralOldestActiveJobIndex OBJECT-TYPE
3515     SYNTAX      Integer32 (0..2147483647)
3516     MAX-ACCESS  read-only
3517     STATUS      current
3518     DESCRIPTION
3519         "The jmJobIndex of the oldest job that is still in one of the
3520         'active' states (pending, processing, or processingStopped).
3521         In other words, the index of the 'active' job that has been in
3522         the job tables the longest.
3523
3524         If there are no active jobs, the agent SHALL set the value of
3525         this object to 0."
3526     REFERENCE
3527         "See Section 3.2 entitled 'The Job Tables and the Oldest Active
3528         and Newest Active Indexes' for a description of the usage of
3529         this object."
3530     DEFVAL      { 0 }          -- no active jobs
3531     ::= { jmGeneralEntry 3 }
3532
3533
3534
3535 jmGeneralNewestActiveJobIndex OBJECT-TYPE
3536     SYNTAX      Integer32 (0..2147483647)
3537     MAX-ACCESS  read-only
3538     STATUS      current
3539     DESCRIPTION
3540         "The jmJobIndex of the newest job that is in one of the
3541         'active' states (pending, processing, or processingStopped).
3542         In other words, the index of the 'active' job that has been
3543         most recently added to the job tables.
3544
3545         When all jobs become 'inactive', i.e., enter the pendingHeld,
3546         completed, canceled, or aborted states, the agent SHALL set the
3547         value of this object to 0."
3548     REFERENCE
3549         "See Section 3.2 entitled 'The Job Tables and the Oldest Active
3550         and Newest Active Indexes' for a description of the usage of
3551         this object."
3552     DEFVAL      { 0 }          -- no active jobs
3553     ::= { jmGeneralEntry 4 }
3554
```

```
3555 jmGeneralJobPersistence OBJECT-TYPE
3556     SYNTAX      Integer32(15..2147483647)
3557     UNITS       "seconds"
3558     MAX-ACCESS  read-only
3559     STATUS      current
3560     DESCRIPTION
3561         "The minimum time in seconds for this instance of the Job Set
3562         that an entry SHALL remain in the jmJobIDTable and jmJobTable
3563         after processing has completed, i.e., the minimum time in
3564         seconds starting when the job enters the completed, canceled,
3565         or aborted state.
3566
3567         Configuring this object is implementation-dependent.
3568
3569         This value SHALL be equal to or greater than the value of
3570         jmGeneralAttributePersistence. This value SHOULD be at least
3571         60 which gives a monitoring application one minute in which to
3572         poll for job data."
3573     DEFVAL      { 60 }          -- one minute
3574     ::= { jmGeneralEntry 5 }
3575
3576
3577
3578 jmGeneralAttributePersistence OBJECT-TYPE
3579     SYNTAX      Integer32(15..2147483647)
3580     UNITS       "seconds"
3581     MAX-ACCESS  read-only
3582     STATUS      current
3583     DESCRIPTION
3584         "The minimum time in seconds for this instance of the Job Set
3585         that an entry SHALL remain in the jmAttributeTable after
3586         processing has completed , i.e., the time in seconds starting
3587         when the job enters the completed, canceled, or aborted state.
3588
3589         Configuring this object is implementation-dependent.
3590
3591         This value SHOULD be at least 60 which gives a monitoring
3592         application one minute in which to poll for job data."
3593     DEFVAL      { 60 }          -- one minute
3594     ::= { jmGeneralEntry 6 }
3595
```

```
3596 jmGeneralJobSetName OBJECT-TYPE
3597     SYNTAX      JmUTF8StringTC(SIZE(0..63))
3598     MAX-ACCESS  read-only
3599     STATUS      current
3600     DESCRIPTION
3601         "The human readable name of this job set assigned by the system
3602         administrator (by means outside of this MIB). Typically, this
3603         name SHOULD be the name of the job queue. If a server or
3604         device has only a single job set, this object can be the
3605         administratively assigned name of the server or device itself.
3606         This name does not need to be unique, though each job set in a
3607         single Job Monitoring MIB SHOULD have distinct names.
3608
3609         NOTE - If the job set corresponds to a single printer and the
3610         Printer MIB is implemented, this value SHOULD be the same as
3611         the prtGeneralPrinterName object in the draft Printer MIB
3612         [print-mib-draft]. If the job set corresponds to an IPP
3613         Printer, this value SHOULD be the same as the IPP 'printer-
3614         name' Printer attribute.
3615
3616         NOTE - The purpose of this object is to help the user of the
3617         job monitoring application distinguish between several job sets
3618         in implementations that support more than one job set."
3619     REFERENCE
3620         "See the OBJECT compliance macro for the minimum maximum length
3621         required for conformance."
3622     DEFVAL      { 'H' }      -- empty string
3623     ::= { jmGeneralEntry 7 }
```

```
3629 -- The Job ID Group (MANDATORY)
3630
3631 -- The jmJobIDGroup consists entirely of the jmJobIDTable.
3632
3633 jmJobID OBJECT IDENTIFIER ::= { jobmonMIBObjects 2 }
3634
3635 jmJobIDTable OBJECT-TYPE
3636     SYNTAX      SEQUENCE OF JmJobIDEntry
3637     MAX-ACCESS  not-accessible
3638     STATUS      current
3639     DESCRIPTION
3640         "The jmJobIDTable provides a correspondence map (1) between the
3641         job submission ID that a client uses to refer to a job and (2)
3642         the jmGeneralJobSetIndex and jmJobIndex that the Job Monitoring
3643         MIB agent assigned to the job and that are used to access the
3644         job in all of the other tables in the MIB.  If a monitoring
3645         application already knows the jmGeneralJobSetIndex and the
3646         jmJobIndex of the job it is querying, that application NEED NOT
3647         use the jmJobIDTable."
3648     REFERENCE
3649         "The MANDATORY-GROUP macro specifies that this group is
3650         MANDATORY."
3651     ::= { jmJobID 1 }
3652
3653
3654
3655 jmJobIDEntry OBJECT-TYPE
3656     SYNTAX      JmJobIDEntry
3657     MAX-ACCESS  not-accessible
3658     STATUS      current
3659     DESCRIPTION
3660         "The map from (1) the jmJobSubmissionID to (2) the
3661         jmGeneralJobSetIndex and jmJobIndex.
3662
3663         An entry SHALL exist in this table for each job currently known
3664         to the agent for all job sets and job states.  There MAY be
3665         more than one jmJobIDEntry that maps to a single job.  This
3666         many to one mapping can occur when more than one network entity
3667         along the job submission path supplies a job submission ID.
3668         See Section 3.5.  However, each job SHALL appear once and in
3669         one and only one job set."
3670     INDEX { jmJobSubmissionID }
3671     ::= { jmJobIDTable 1 }
3672
3673 JmJobIDEntry ::= SEQUENCE {
3674     jmJobSubmissionID          OCTET STRING(SIZE(48)),
3675     jmJobIDJobSetIndex        Integer32(0..32767),
3676     jmJobIDJobIndex           Integer32(0..2147483647)
3677 }
3678
```

3679 jmJobSubmissionID OBJECT-TYPE
3680 SYNTAX OCTET STRING(SIZE(48))
3681 MAX-ACCESS not-accessible
3682 STATUS current
3683 DESCRIPTION
3684 "A quasi-unique 48-octet fixed-length string ID which
3685 identifies the job within a particular client-server
3686 environment. There are multiple formats for the
3687 jmJobSubmissionID. Each format SHALL be uniquely identified.
3688 See the JmJobSubmissionIDTypeTC textual convention. Each
3689 format SHALL be registered using the procedures of a type 2
3690 enum. See section 3.7.3 entitled: 'PWG Registration of Job
3691 Submission Id Formats'.
3692
3693 If the requester (client or server) does not supply a job
3694 submission ID in the job submission protocol, then the
3695 recipient (server or device) SHALL assign a job submission ID
3696 using any of the standard formats that have been reserved for
3697 agents and adding the final 8 octets to distinguish the ID from
3698 others submitted from the same requester.
3699
3700 The monitoring application, whether in the client or running
3701 separately, MAY use the job submission ID to help identify
3702 which jmJobIndex was assigned by the agent, i.e., in which row
3703 the job information is in the other tables.
3704
3705 NOTE - fixed-length is used so that a management application
3706 can use a shortened GetNext varbind (in SNMPv1 and SNMPv2) in
3707 order to get the next submission ID, disregarding the remainder
3708 of the ID in order to access jobs independent of the trailing
3709 identifier part, e.g., to get all jobs submitted by a
3710 particular jmJobOwner or submitted from a particular MAC
3711 address."
3712 REFERENCE
3713 "See the JmJobSubmissionIDTypeTC textual convention.
3714 See APPENDIX B - Support of Job Submission Protocols."
3715 ::= { jmJobIDEntry 1 }
3716

```
3717 jmJobIDJobSetIndex OBJECT-TYPE
3718     SYNTAX      Integer32(0..32767)
3719     MAX-ACCESS  read-only
3720     STATUS      current
3721     DESCRIPTION
3722         "This object contains the value of the jmGeneralJobSetIndex for
3723         the job with the jmJobSubmissionID value, i.e., the job set
3724         index of the job set in which the job was placed when that
3725         server or device accepted the job.  This 16-bit value in
3726         combination with the jmJobIDJobIndex value permits the
3727         management application to access the other tables to obtain the
3728         job-specific objects for this job."
3729     REFERENCE
3730         "See jmGeneralJobSetIndex in the jmGeneralTable."
3731     DEFVAL      { 0 }      -- 0 indicates no job set index
3732     ::= { jmJobIDEntry 2 }
3733
3734
3735
3736 jmJobIDJobIndex OBJECT-TYPE
3737     SYNTAX      Integer32(0..2147483647)
3738     MAX-ACCESS  read-only
3739     STATUS      current
3740     DESCRIPTION
3741         "This object contains the value of the jmJobIndex for the job
3742         with the jmJobSubmissionID value, i.e., the job index for the
3743         job when the server or device accepted the job.  This value, in
3744         combination with the jmJobIDJobSetIndex value, permits the
3745         management application to access the other tables to obtain the
3746         job-specific objects for this job."
3747     REFERENCE
3748         "See jmJobIndex in the jmJobTable."
3749     DEFVAL      { 0 }      -- 0 indicates no jmJobIndex value.
3750     ::= { jmJobIDEntry 3 }
3751
3752
3753
3754
```



```

3755 -- The Job Group (MANDATORY)
3756
3757 -- The jmJobGroup consists entirely of the jmJobTable.
3758
3759 jmJob OBJECT IDENTIFIER ::= { jobmonMIBObjects 3 }
3760
3761 jmJobTable OBJECT-TYPE
3762     SYNTAX      SEQUENCE OF JmJobEntry
3763     MAX-ACCESS  not-accessible
3764     STATUS      current
3765     DESCRIPTION
3766         "The jmJobTable consists of basic job state and status
3767         information for each job in a job set that (1) monitoring
3768         applications need to be able to access in a single SNMP Get
3769         operation, (2) that have a single value per job, and (3) that
3770         SHALL always be implemented."
3771     REFERENCE
3772         "The MANDATORY-GROUP macro specifies that this group is
3773         MANDATORY."
3774     ::= { jmJob 1 }
3775
3776
3777
3778 jmJobEntry OBJECT-TYPE
3779     SYNTAX      JmJobEntry
3780     MAX-ACCESS  not-accessible
3781     STATUS      current
3782     DESCRIPTION
3783         "Basic per-job state and status information.
3784
3785         An entry SHALL exist in this table for each job, no matter what
3786         the state of the job is. Each job SHALL appear in one and only
3787         one job set."
3788     REFERENCE
3789         "See Section 3.2 entitled 'The Job Tables'."
3790     INDEX { jmGeneralJobSetIndex, jmJobIndex }
3791     ::= { jmJobTable 1 }
3792
3793 JmJobEntry ::= SEQUENCE {
3794     jmJobIndex          Integer32(1..2147483647),
3795     jmJobState          JmJobStateTC,
3796     jmJobStateReasons1 JmJobStateReasons1TC,
3797     jmNumberOfInterveningJobs Integer32(-2..2147483647),
3798     jmJobKOctetsPerCopyRequested Integer32(-2..2147483647),
3799     jmJobKOctetsProcessed Integer32(-2..2147483647),
3800     jmJobImpressionsPerCopyRequested Integer32(-2..2147483647),
3801     jmJobImpressionsCompleted Integer32(-2..2147483647),
3802     jmJobOwner          JmJobStringTC(SIZE(0..63))
3803 }
3804

```

```
3805 jmJobIndex OBJECT-TYPE
3806     SYNTAX      Integer32(1..2147483647)
3807     MAX-ACCESS  not-accessible
3808     STATUS      current
3809     DESCRIPTION
3810         "The sequential, monotonically increasing identifier index for
3811         the job generated by the server or device when that server or
3812         device accepted the job.  This index value permits the
3813         management application to access the other tables to obtain the
3814         job-specific row entries."
3815     REFERENCE
3816         "See Section 3.2 entitled 'The Job Tables and the Oldest Active
3817         and Newest Active Indexes'.
3818         See Section 3.5 entitled 'Job Identification'.
3819         See also
3820
3821         jmGeneralNewestActiveJobIndex for the largest value of
3822         jmJobIndex.
3823         See JmJobSubmissionIDTypeTC for a limit on the size of this
3824         index if the agent represents it as an 8-digit decimal number."
3825     ::= { jmJobEntry 1 }
3826
3827
3828
3829 jmJobState OBJECT-TYPE
3830     SYNTAX      JmJobStateTC
3831     MAX-ACCESS  read-only
3832     STATUS      current
3833     DESCRIPTION
3834         "The current state of the job (pending, processing, completed,
3835         etc.).  Agents SHALL implement only those states which are
3836         appropriate for the particular implementation.  However,
3837         management applications SHALL be prepared to receive all the
3838         standard job states.
3839
3840         The final value for this object SHALL be one of: completed,
3841         canceled, or aborted.  The minimum length of time that the
3842         agent SHALL maintain MIB data for a job in the completed,
3843         canceled, or aborted state before removing the job data from
3844         the jmJobIDTable and jmJobTable is specified by the value of
3845         the jmGeneralJobPersistence object."
3846     DEFVAL      { unknown }          -- default is unknown
3847     ::= { jmJobEntry 2 }
3848
```

```
3849 jmJobStateReasons1 OBJECT-TYPE
3850     SYNTAX      JmJobStateReasons1TC
3851     MAX-ACCESS  read-only
3852     STATUS      current
3853     DESCRIPTION
3854         "Additional information about the job's current state, i.e.,
3855         information that augments the value of the job's jmJobState
3856         object.
3857
3858         Implementation of any reason values is OPTIONAL, but an agent
3859         SHOULD return any reason information available. These values
3860         MAY be used with any job state or states for which the reason
3861         makes sense. Since the Job State Reasons will be more dynamic
3862         than the Job State, it is recommended that a job monitoring
3863         application read this object every time jmJobState is read.
3864         When the agent cannot provide a reason for the current state of
3865         the job, the value of the jmJobStateReasons1 object and
3866         jobStateReasonsN attributes SHALL be 0."
3867     REFERENCE
3868         "The jobStateReasonsN (N=2..4) attributes provide further
3869         additional information about the job's current state."
3870     DEFVAL      { 0 }          -- no reasons
3871     ::= { jmJobEntry 3 }
3872
3873
3874
3875 jmNumberOfInterveningJobs OBJECT-TYPE
3876     SYNTAX      Integer32(-2..2147483647)
3877     MAX-ACCESS  read-only
3878     STATUS      current
3879     DESCRIPTION
3880         "The number of jobs that are expected to complete processing
3881         before this job has completed processing according to the
3882         implementation's queuing algorithm, if no other jobs were to be
3883         submitted. In other words, this value is the job's queue
3884         position. The agent SHALL return a value of 0 for this
3885         attribute when the job is the next job to complete processing
3886         (or has completed processing)."
3887     DEFVAL      { 0 }          -- default is no intervening jobs.
3888     ::= { jmJobEntry 4 }
3889
```

```
3890 jmJobKOctetsPerCopyRequested OBJECT-TYPE
3891     SYNTAX      Integer32(-2..2147483647)
3892     MAX-ACCESS  read-only
3893     STATUS      current
3894     DESCRIPTION
3895         "The total size in K (1024) octets of the document(s) being
3896         requested to be processed in the job.  The agent SHALL round
3897         the actual number of octets up to the next highest K.  Thus 0
3898         octets SHALL be represented as '0', 1-1024 octets SHALL be
3899         represented as '1', 1025-2048 SHALL be represented as '2', etc.
3900
3901         In computing this value, the server/device SHALL not include
3902         the multiplicative factors contributed by (1) the number of
3903         document copies, and (2) the number of job copies, independent
3904         of whether the device can process multiple copies of the job or
3905         document without making multiple passes over the job or
3906         document data and independent of whether the output is collated
3907         or not.  Thus the server/device computation is independent of
3908         the implementation and indicates the size of the document(s)
3909         measured in K octets independent of the number of copies."
3910     DEFVAL      { -2 }      -- the default is unknown(-2)
3911     ::= { jmJobEntry 5 }
3912
3913
3914
3915 jmJobKOctetsProcessed OBJECT-TYPE
3916     SYNTAX      Integer32(-2..2147483647)
3917     MAX-ACCESS  read-only
3918     STATUS      current
3919     DESCRIPTION
3920         "The total number of octets processed by the server or device
3921         measured in units of K (1024) octets so far.  The agent SHALL
3922         round the actual number of octets processed up to the next
3923         higher K.  Thus 0 octets SHALL be represented as '0', 1-1024
3924         octets SHALL be represented as '1', 1025-2048 octets SHALL be
3925         '2', etc.  For printing devices, this value is the number
3926         interpreted by the page description language interpreter rather
3927         than what has been marked on media.
3928
3929         For implementations where multiple copies are produced by the
3930         interpreter with only a single pass over the data, the final
3931         value SHALL be equal to the value of the
3932         jmJobKOctetsPerCopyRequested object.  For implementations where
3933         multiple copies are produced by the interpreter by processing
3934         the data for each copy, the final value SHALL be a multiple of
3935         the value of the jmJobKOctetsPerCopyRequested object.
3936
3937         NOTE - See the impressionsCompletedCurrentCopy and
3938         pagesCompletedCurrentCopy attributes for attributes that are
3939         reset on each document copy.
3940
```

3941 NOTE - The jmJobKOctetsProcessed object can be used with the
3942 jmJobKOctetsPerCopyRequested object to provide an indication of
3943 the relative progress of the job, provided that the
3944 multiplicative factor is taken into account for some
3945 implementations of multiple copies."
3946 DEFVAL { 0 } -- default is no octets processed.
3947 ::= { jmJobEntry 6 }
3948
3949
3950 jmJobImpressionsPerCopyRequested OBJECT-TYPE
3951 SYNTAX Integer32(-2..2147483647)
3952 MAX-ACCESS read-only
3953 STATUS current
3954 DESCRIPTION
3955 "The total size in number of impressions of the document(s)
3956 submitted.
3957
3958 In computing this value, the server/device SHALL not include
3959 the multiplicative factors contributed by (1) the number of
3960 document copies, and (2) the number of job copies, independent
3961 of whether the device can process multiple copies of the job or
3962 document without making multiple passes over the job or
3963 document data and independent of whether the output is collated
3964 or not. Thus the server/device computation is independent of
3965 the implementation and reflects the size of the document(s)
3966 measured in impressions independent of the number of copies."
3967 REFERENCE
3968 "See the definition of the term 'impression' in Section 2."
3969 DEFVAL { -2 } -- default is unknown(-2)
3970 ::= { jmJobEntry 7 }
3971
3972
3973 jmJobImpressionsCompleted OBJECT-TYPE
3974 SYNTAX Integer32(-2..2147483647)
3975 MAX-ACCESS read-only
3976 STATUS current
3977 DESCRIPTION
3978 "The total number of impressions completed for this job so far.
3979 For printing devices, the impressions completed includes
3980 interpreting, marking, and stacking the output. For other
3981 types of job services, the number of impressions completed
3982 includes the number of impressions processed.
3983
3984 NOTE - See the impressionsCompletedCurrentCopy and
3985 pagesCompletedCurrentCopy attributes for attributes that are
3986 reset on each document copy.
3987
3988 NOTE - The jmJobImpressionsCompleted object can be used with
3989 the jmJobImpressionsPerCopyRequested object to provide an
3990 indication of the relative progress of the job, provided that
3991 the multiplicative factor is taken into account for some
3992 implementations of multiple copies."

```
3993     REFERENCE
3994         "See the definition of the term 'impression' in Section 2 and
3995         the counting example in Section 3.4 entitled 'Monitoring Job
3996         Progress'."
3997     DEFVAL      { 0 }          -- default is no octets
3998     ::= { jmJobEntry 8 }
3999
4000
4001
4002 jmJobOwner OBJECT-TYPE
4003     SYNTAX      JmJobStringTC(SIZE(0..63))
4004     MAX-ACCESS  read-only
4005     STATUS      current
4006     DESCRIPTION
4007         "The coded character set name of the user that submitted the
4008         job.  The method of assigning this user name will be system
4009         and/or site specific but the method MUST insure that the name
4010         is unique to the network that is visible to the client and
4011         target device.
4012
4013         This value SHOULD be the most authenticated name of the user
4014         submitting the job."
4015     REFERENCE
4016         "See the OBJECT compliance macro for the minimum maximum length
4017         required for conformance."
4018     DEFVAL      { ''H }          -- empty string
4019     ::= { jmJobEntry 9 }
4020
4021
4022
4023
```

```
4024 -- The Attribute Group (MANDATORY)
4025
4026 -- The jmAttributeGroup consists entirely of the jmAttributeTable.
4027 --
4028 -- Implementation of the two objects in this group is MANDATORY.
4029 -- See Section 3.1 entitled 'Conformance Considerations'.
4030 -- An agent SHALL implement any attribute if (1) the server or device
4031 -- supports the functionality represented by the attribute and (2) the
4032 -- information is available to the agent.
4033
4034 jmAttribute OBJECT IDENTIFIER ::= { jobmonMIBObjects 4 }
4035
4036
4037
4038 jmAttributeTable OBJECT-TYPE
4039     SYNTAX          SEQUENCE OF JmAttributeEntry
4040     MAX-ACCESS      not-accessible
4041     STATUS          current
4042     DESCRIPTION
4043         "The jmAttributeTable SHALL contain attributes of the job and
4044         document(s) for each job in a job set.  Instead of allocating
4045         distinct objects for each attribute, each attribute is
4046         represented as a separate row in the jmAttributeTable."
4047     REFERENCE
4048         "The MANDATORY-GROUP macro specifies that this group is
4049         MANDATORY.  An agent SHALL implement any attribute if (1) the
4050         server or device supports the functionality represented by the
4051         attribute and (2) the information is available to the agent. "
4052     ::= { jmAttribute 1 }
4053
4054
4055
4056 jmAttributeEntry OBJECT-TYPE
4057     SYNTAX          JmAttributeEntry
4058     MAX-ACCESS      not-accessible
4059     STATUS          current
4060     DESCRIPTION
4061         "Attributes representing information about the job and
4062         document(s) or resources required and/or consumed.
4063
4064         Each entry in the jmAttributeTable is a per-job entry with an
4065         extra index for each type of attribute (jmAttributeTypeIndex)
4066         that a job can have and an additional index
4067         (jmAttributeInstanceIndex) for those attributes that can have
4068         multiple instances per job.  The jmAttributeTypeIndex object
4069         SHALL contain an enum type that indicates the type of attribute
4070         (see the JmAttributeTypeTC textual-convention).  The value of
4071         the attribute SHALL be represented in either the
4072         jmAttributeValueAsInteger or jmAttributeValueAsOctets objects,
4073         and/or both, as specified in the JmAttributeTypeTC textual-
4074         convention.
4075
```

4076 The agent SHALL create rows in the jmAttributeTable as the
 4077 server or device is able to discover the attributes either from
 4078 the job submission protocol itself or from the document PDL.
 4079 As the documents are interpreted, the interpreter MAY discover
 4080 additional attributes and so the agent adds additional rows to
 4081 this table. As the attributes that represent resources are
 4082 actually consumed, the usage counter contained in the
 4083 jmAttributeValueAsInteger object is incremented according to
 4084 the units indicated in the description of the JmAttributeTypeTC
 4085 enum.

4087 The agent SHALL maintain each row in the jmJobTable for at
 4088 least the minimum time after a job completes as specified by
 4089 the jmGeneralAttributePersistence object.

4091 Zero or more entries SHALL exist in this table for each job in
 4092 a job set."

4093 REFERENCE

4094 "See Section 3.3 entitled 'The Attribute Mechanism' for a
 4095 description of the jmAttributeTable."

4096 INDEX { jmGeneralJobSetIndex, jmJobIndex, jmAttributeTypeIndex,
 4097 jmAttributeInstanceIndex }
 4098 ::= { jmAttributeTable 1 }

4100 JmAttributeEntry ::= SEQUENCE {
 4101 jmAttributeTypeIndex JmAttributeTypeTC,
 4102 jmAttributeInstanceIndex Integer32(1..32767),
 4103 jmAttributeValueAsInteger Integer32(-2..2147483647),
 4104 jmAttributeValueAsOctets OCTET STRING(SIZE(0..63))
 4105 }
 4106


```
4107 jmAttributeTypeIndex OBJECT-TYPE
4108     SYNTAX      JmAttributeTypeTC
4109     MAX-ACCESS  not-accessible
4110     STATUS      current
4111     DESCRIPTION
4112         "The type of attribute that this row entry represents.
4113
4114         The type MAY identify information about the job or document(s)
4115         or MAY identify a resource required to process the job before
4116         the job start processing and/or consumed by the job as the job
4117         is processed.
4118
4119         Examples of job attributes (i.e., apply to the job as a whole)
4120         that have only one instance per job include:
4121         jobCopiesRequested(90), documentCopiesRequested(92),
4122         jobCopiesCompleted(91), documentCopiesCompleted(93), while
4123         examples of job attributes that may have more than one instance
4124         per job include: documentFormatIndex(37), and
4125         documentFormat(38).
4126
4127         Examples of document attributes (one instance per document)
4128         include: fileName(34), and documentName(35).
4129
4130         Examples of required and consumed resource attributes include:
4131         pagesRequested(130), mediumRequested(170), pagesCompleted(131),
4132         and mediumConsumed(171), respectively."
4133 ::= { jmAttributeEntry 1 }
4134
4135
4136
4137 jmAttributeInstanceIndex OBJECT-TYPE
4138     SYNTAX      Integer32(1..32767)
4139     MAX-ACCESS  not-accessible
4140     STATUS      current
4141     DESCRIPTION
4142         "A running 16-bit index of the attributes of the same type for
4143         each job.  For those attributes with only a single instance per
4144         job, this index value SHALL be 1.  For those attributes that
4145         are a single value per document, the index value SHALL be the
4146         document number, starting with 1 for the first document in the
4147         job.  Jobs with only a single document SHALL use the index
4148         value of 1.  For those attributes that can have multiple values
4149         per job or per document, such as documentFormatIndex(37) or
4150         documentFormat(38), the index SHALL be a running index for the
4151         job as a whole, starting at 1."
4152 ::= { jmAttributeEntry 2 }
4153
```

```

4154 jmAttributeValueAsInteger OBJECT-TYPE
4155     SYNTAX      Integer32(-2..2147483647)
4156     MAX-ACCESS  read-only
4157     STATUS      current
4158     DESCRIPTION
4159         "The integer value of the attribute.  The value of the
4160         attribute SHALL be represented as an integer if the enum
4161         description in the JmAttributeTypeTC textual-convention
4162         definition has the tag: 'INTEGER:'.
4163
4164         Depending on the enum definition, this object value MAY be an
4165         integer, a counter, an index, or an enum, depending on the
4166         jmAttributeTypeIndex value.  The units of this value are
4167         specified in the enum description.
4168
4169         For those attributes that are accumulating job consumption as
4170         the job is processed as specified in the JmAttributeTypeTC
4171         textual-convention, SHALL contain the final value after the job
4172         completes processing, i.e., this value SHALL indicate the total
4173         usage of this resource made by the job.
4174
4175         A monitoring application is able to copy this value to a
4176         suitable longer term storage for later processing as part of an
4177         accounting system.
4178
4179         Since the agent MAY add attributes representing resources to
4180         this table while the job is waiting to be processed or being
4181         processed, which can be a long time before any of the resources
4182         are actually used, the agent SHALL set the value of the
4183         jmAttributeValueAsInteger object to 0 for resources that the
4184         job has not yet consumed.
4185
4186         Attributes for which the concept of an integer value is
4187         meaningless, such as fileName(34), jobName, and
4188         processingMessage, do not have the 'INTEGER:' tag in the
4189         JmAttributeTypeTC definition and so an agent SHALL always
4190         return a value of '-1' to indicate 'other' for the value of the
4191         jmAttributeValueAsInteger object for these attributes.
4192
4193         For attributes which do have the 'INTEGER:' tag in the
4194         JmAttributeTypeTC definition, if the integer value is not (yet)
4195         known, the agent either (1) SHALL not materialize the row in
4196         the jmAttributeTable until the value is known or (2) SHALL
4197         return a '-2' to represent an 'unknown' counting integer value,
4198         a '0' to represent an 'unknown' index value, and a '2' to
4199         represent an 'unknown(2)' enum value."
4200     DEFVAL      { -2 }      -- default value is unknown(-2)
4201     ::= { jmAttributeEntry 3 }
4202

```

```
4203 jmAttributeValueAsOctets OBJECT-TYPE
4204     SYNTAX      OCTET STRING(SIZE(0..63))
4205     MAX-ACCESS  read-only
4206     STATUS      current
4207     DESCRIPTION
4208         "The octet string value of the attribute.  The value of the
4209         attribute SHALL be represented as an OCTET STRING if the enum
4210         description in the JmAttributeTypeTC textual-convention
4211         definition has the tag: 'OCTETS:'.
4212
4213         Depending on the enum definition, this object value MAY be a
4214         coded character set string (text), such as 'JmUTF8StringTC', or
4215         a binary octet string, such as 'DateAndTime'.
4216
4217         Attributes for which the concept of an octet string value is
4218         meaningless, such as pagesCompleted, do not have the tag
4219         'OCTETS:' in the JmAttributeTypeTC definition and so the agent
4220         SHALL always return a zero length string for the value of the
4221         jmAttributeValueAsOctets object.
4222
4223         For attributes which do have the 'OCTETS:' tag in the
4224         JmAttributeTypeTC definition, if the OCTET STRING value is not
4225         (yet) known, the agent either SHALL not materialize the row in
4226         the jmAttributeTable until the value is known or SHALL return a
4227         zero-length string."
4228     DEFVAL      { 'H' } -- empty string
4229     ::= { jmAttributeEntry 4 }
4230
```

```
4231 -- Notifications and Trapping
4232 -- Reserved for the future
4233
4234 jobmonMIBNotifications OBJECT IDENTIFIER ::= { jobmonMIB 2}
4235
4236
4237
4238 -- Conformance Information
4239
4240 jmMIBConformance OBJECT IDENTIFIER ::= { jobmonMIB 3 }
4241
4242
4243
4244 -- compliance statements
4245 jmMIBCompliance MODULE-COMPLIANCE
4246     STATUS current
4247     DESCRIPTION
4248         "The compliance statement for agents that implement the
4249         job monitoring MIB."
4250     MODULE -- this module
4251     MANDATORY-GROUPS {
4252         jmGeneralGroup, jmJobIDGroup, jmJobGroup, jmAttributeGroup }
4253
4254     OBJECT jmGeneralJobSetName
4255     SYNTAX JmUTF8StringTC (SIZE(0..8))
4256     DESCRIPTION
4257         "Only 8 octets maximum string length NEED be supported by the
4258         agent."
4259
4260     OBJECT jmJobOwner
4261     SYNTAX JmJobStringTC (SIZE(0..16))
4262     DESCRIPTION
4263         "Only 16 octets maximum string length NEED be supported by the
4264         agent."
4265
4266 -- There are no CONDITIONALLY MANDATORY or OPTIONAL groups.
4267
4268 ::= { jmMIBConformance 1 }
4269
```

```
4270 jmMIBGroups      OBJECT IDENTIFIER ::= { jmMIBConformance 2 }
4271
4272 jmGeneralGroup OBJECT-GROUP
4273     OBJECTS {
4274         jmGeneralNumberOfActiveJobs,    jmGeneralOldestActiveJobIndex,
4275         jmGeneralNewestActiveJobIndex,   jmGeneralJobPersistence,
4276         jmGeneralAttributePersistence,   jmGeneralJobSetName}
4277     STATUS current
4278     DESCRIPTION
4279         "The general group."
4280     ::= { jmMIBGroups 1 }
4281
4282
4283
4284 jmJobIDGroup OBJECT-GROUP
4285     OBJECTS {
4286         jmJobIDJobSetIndex, jmJobIDJobIndex }
4287     STATUS current
4288     DESCRIPTION
4289         "The job ID group."
4290     ::= { jmMIBGroups 2 }
4291
4292
4293
4294 jmJobGroup OBJECT-GROUP
4295     OBJECTS {
4296         jmJobState, jmJobStateReasons1, jmNumberOfInterveningJobs,
4297         jmJobKOctetsPerCopyRequested, jmJobKOctetsProcessed,
4298         jmJobImpressionsPerCopyRequested, jmJobImpressionsCompleted,
4299         jmJobOwner }
4300     STATUS current
4301     DESCRIPTION
4302         "The job group."
4303     ::= { jmMIBGroups 3 }
4304
4305
4306
4307 jmAttributeGroup OBJECT-GROUP
4308     OBJECTS {
4309         jmAttributeValueAsInteger, jmAttributeValueAsOctets }
4310     STATUS current
4311     DESCRIPTION
4312         "The attribute group."
4313     ::= { jmMIBGroups 4 }
4314
4315
4316 END
```

4317 5. Appendix A - Implementing the Job Life Cycle

4318 The job object has well-defined states and client operations that
4319 affect the transition between the job states. Internal server and
4320 device actions also affect the transitions of the job between the job
4321 states. These states and transitions are referred to as the job's *life*
4322 *cycle*.

4323 Not all implementations of job submission protocols have all of the
4324 states of the job model specified here. The job model specified here
4325 is intended to be a superset of most implementations. It is the
4326 purpose of the agent to map the particular implementation's job life
4327 cycle onto the one specified here. The agent MAY omit any states not
4328 implemented. Only the processing and completed states are required to
4329 be implemented by an agent. However, a conforming management
4330 application SHALL be prepared to accept any of the states in the job
4331 life cycle specified here, so that the management application can
4332 interoperate with any conforming agent.

4333 The job states are intended to be user visible. The agent SHALL make
4334 these states visible in the MIB, but only for the subset of job states
4335 that the implementation has. Some implementations MAY need to have
4336 sub-states of these user-visible states. The jmJobStateReasons1 object
4337 and the jobStateReasonsN ($N=2..4$) attributes can be used to represent
4338 the sub-states of the jobs.

4339 Job states are intended to last a user-visible length of time in most
4340 implementations. However, some jobs may pass through some states in
4341 zero time in some situations and/or in some implementations.

4342 The job model does not specify how accounting and auditing is
4343 implemented, except to assume that accounting and auditing logs are
4344 separate from the job life cycle and last longer than job entries in
4345 the MIB. Jobs in the completed, aborted, or canceled states are not
4346 logs, since jobs in these states are accessible via SNMP protocol
4347 operations and SHALL be removed from the Job Monitoring MIB tables
4348 after a site-settable or implementation-defined period of time. An
4349 accounting application MAY copy accounting information incrementally to
4350 an accounting log as a job processes, or MAY be copied while the job is
4351 in the canceled, aborted, or completed states, depending on
4352 implementation. The same is true for auditing logs.

4353 The jmJobState object specifies the standard job states. The normal
4354 job state transitions are shown in the state transition diagram
4355 presented in Table 1.

4356 6. APPENDIX B - Support of Job Submission Protocols

4357 A companion PWG document, entitled "Job Submission Protocol Mapping
4358 Recommendations for the Job Monitoring MIB" [protomap] contains the
4359 recommended usage of each of the objects and attributes in this MIB
4360 with a number of job submission protocols. In particular, which job
4361 submission ID format should be used is indicated for each job
4362 submission protocol.

4363 Some job submission protocols have support for the client to specify a
4364 job submission ID. A second approach is to enhance the document format
4365 to embed the job submission ID in the document data. This second
4366 approach is independent of the job submission protocol. This appendix
4367 lists some examples of these approaches.

4368 Some PJL implementations wrap a banner page as a PJL job around a job
4369 submitted by a client. If this results in multiple job submission IDs,
4370 the agent SHALL create multiple jmJobIDEntry rows in the jmJobIDTable
4371 that each point to the same job entry in the job tables. See the
4372 specification of the jmJobIDEntry.

4373 7. References

4374 [char-set policy] Harald Avelstrand, "IETF Policy on Character Sets and
4375 Language", June 1997. Latest draft: <draft-avelstrand-charset-
4376 policy-00.txt>

4377 [GB2312] GB 2312-1980, "Chinese People's Republic of China (PRC) mixed
4378 one byte and two byte coded character set"

4379 [hr-mib] P. Grillo, S. Waldbusser, "Host Resources MIB", RFC 1514,
4380 September 1993

4381 [iana] J. Reynolds, and J. Postel, "Assigned Numbers", STD 2, RFC 1700,
4382 ISI, October 1994.

4383 [IANA-charsets] Coded Character Sets registered by IANA and assigned an
4384 enum value for use in the CodedCharSet textual convention imported from
4385 the Printer MIB. See ftp://ftp.isi.edu/in-
4386 notes/iana/assignments/character-sets

4387 [iana-media-types] IANA Registration of MIME media types (MIME content
4388 types/subtypes). See ftp://ftp.isi.edu/in-notes/iana/assignments/

4389 [ISO-639] ISO 639:1988 (E/F) - Code for Representation of names of
4390 languages - The International Organization for Standardization, 1st
4391 edition, 1988.

4392 [ISO 646] ISO/IEC 646:1991, "Information technology -- ISO 7-bit coded
4393 character set for information interchange", JTC1/SC2.

- 4394 [ISO 8859] ISO/IEC 8859-1:1987, "Information technology -- 8-bit single
4395 byte coded graphic character sets - Part 1: Latin alphabet No. 1,
4396 JTC1/SC2."
- 4397 [ISO 2022] ISO/IEC 2022:1994 - "Information technology -- Character
4398 code structure and extension techniques", JTC1/SC2.
- 4399 [ISO-3166] ISO 3166:1988 (E/F) - Codes for representation of names of
4400 countries - The International Organization for Standardization, 3rd
4401 edition, 1988-08-15."
- 4402 [ISO-10646] ISO/IEC 10646-1:1993, "Information technology -- Universal
4403 Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and
4404 Basic Multilingual Plane, JTC1/SC2.
- 4405 [iso-dpa] ISO/IEC 10175 Document Printing Application (DPA). See
4406 <ftp://ftp.pwg.org/pub/pwg/dpa/>
- 4407 [ipp-model] Internet Printing Protocol/1.0: Model and Semantics, work
4408 in progress on the IETF standards track. See draft-ietf-ipp-model-
4409 09.txt. See also <http://www.pwg.org/ipp/index.html>
- 4410 [JIS X0208] JIS X0208-1990, "Japanese two byte coded character set."
- 4411 [mib-II] MIB-II, RFC 1213.
- 4412 [print-mib] Smith, R., Wright, F., Hastings, T., Zilles, S. and
4413 Gyllenskog, J., "Printer MIB", RFC 1759, proposed IETF standard, March
4414 1995. See also [draft-print-mib].
- 4415 [print-mib-draft] Turner, R., "Printer MIB", work in progress, on the
4416 standards track as a draft standard: <draft-ietf-printmib-mib-info-
4417 02.txt>, October 15, 1997.
- 4418 [protomap] Bergman, R., "Job Submission Protocol Mapping
4419 Recommendations for the Job Monitoring MIB," work in progress as an
4420 informational RFC. See <draft-bergman-printmib-job-protomap-01.txt>,
4421 January 12, 1998.
- 4422 [pwg] The Printer Working Group is a printer industry consortium open
4423 to any individuals. For more information, access the PWG web page:
4424 <http://www.pwg.org>
- 4425 [req-words] S. Bradner, "Keywords for use in RFCs to Indicate
4426 Requirement Levels", RFC 2119, March 1997.
- 4427 [rfc 1738] Berners-Lee, T., Masinter, L., McCahill, M., "Uniform
4428 Resource Locators (URL)", RFC 1738, December 1994.
- 4429 [RFC-1766] Avelstrand, H., "Tags for the Identification of Languages",
4430 RFC 1766, March 1995.

- 4431 [rfc 2130] C. Weider, C. Preston, K. Simonsen, H. Alvestrand, R.
4432 Atkinson, M. Crispin, and P. Svanberg, "The Report of the IAB Character
4433 Set Workshop held 29 Feb-1 March, 1997", April 1997, RFC 2130.
- 4434 [SMIV2-SMI] J. Case, et al. "Structure of Management Information for
4435 Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC
4436 1902, January 1996.
- 4437 [SMIV2-TC] J. Case, et al. "Textual Conventions for Version 2 of the
4438 Simple Network Management Protocol (SNMPv2)", RFC 1903, January 1996.
- 4439 [tipsi] IEEE 1284.1, Transport-independent Printer System Interface
4440 (TIPSI).
- 4441 [URI-spec] Berners-Lee, T., Masinter, L., McCahill, M. , "Uniform
4442 Resource Locators (URL)", RFC 1738, December, 1994.
- 4443 [US-ASCII] Coded Character Set - 7-bit American Standard Code for
4444 Information Interchange, ANSI X3.4-1986.
- 4445 [UTF-8] F. Yergeau, "UTF-8, a transformation format of Unicode and ISO
4446 10646", RFC 2044, October 1996.

4447 8. Author's Addresses

- 4448 Ron Bergman
4449 Dataproducts Corp.
4450 1757 Tapo Canyon Road
4451 Simi Valley, CA 93063-3394
4452
4453 Phone: 805-578-4421
4454 Fax: 805-578-4001
4455 Email: rbergman@dpc.com
4456
4457
4458 Tom Hastings
4459 Xerox Corporation, ESAE-231
4460 701 S. Aviation Blvd.
4461 El Segundo, CA 90245
4462
4463 Phone: 310-333-6413
4464 Fax: 310-333-5514
4465 EMail: hastings@cpl0.es.xerox.com
4466
4467
4468 Scott A. Isaacson
4469 Novell, Inc.
4470 122 E 1700 S
4471 Provo, UT 84606
4472
4473 Phone: 801-861-7366
4474 Fax: 801-861-4025
4475 EMail: scott_isaacson@novell.com

4476

4477

4478

Harry Lewis

4479

IBM Corporation

4480

6300 Diagonal Hwy

4481

Boulder, CO 80301

4482

4483

Phone: (303) 924-5337

4484

Fax:

4485

Email: harryl@us.ibm.com

4486

4487

4488

Send questions and comments to the Printer Working Group (PWG)

4489

using the Job Monitoring Project (JMP) Mailing List: jmp@pwg.org

4490

4491

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

4492

4493

Implementers of this specification are encouraged to join the jmp

4494

mailing list in order to participate in discussions on any

4495

clarifications needed and registration proposals for additional

4496

attributes and values being reviewed in order to achieve consensus.

4497

4498

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

4499

4500

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

4501

4502

Other Participants:

4503

Chuck Adams - Tektronix

4504

Jeff Barnett - IBM

4505

Keith Carter, IBM Corporation

4506

Jeff Copeland - QMS

4507

Andy Davidson - Tektronix

4508

Roger deBry - IBM

4509

Mabry Dozier - QMS

4510

Lee Ferrel - Canon

4511

Steve Gebert - IBM

4512

Robert Herriot - Sun Microsystems Inc.

4513

Shige Kanemitsu - Kyocera

4514

David Kellerman - Northlake Software

4515

Rick Landau - Digital

4516

Pete Loya - HP

4517

Ray Lutz - Cognisys

4518

Jay Martin - Underscore

4519

Mike MacKay, Novell, Inc.

4520

Stan McConnell - Xerox

4521

Carl-Uno Manros, Xerox, Corp.

4522

Pat Nogay - IBM

4523

Bob Pentecost - HP

4524

Rob Rhoads - Intel

4525

David Roach - Unisys

4526

Stuart Rowley - Kyocera

4527 Hiroyuki Sato - Canon
4528 Bob Setterbo - Adobe
4529 Gail Songer, EFI
4530 Mike Timperman - Lexmark
4531 Randy Turner - Sharp
4532 William Wagner - Digital Products
4533 Jim Walker - Dazel
4534 Chris Wellens - Interworking Labs
4535 Rob Whittle - Novell
4536 Don Wright - Lexmark
4537 Lloyd Young - Lexmark
4538 Atsushi Yuki - Kyocera
4539 Peter Zehler, Xerox, Corp.

4540 9. INDEX

4541 This index includes the textual conventions, the objects, and the
 4542 attributes. Textual conventions all start with the prefix: "JM" and
 4543 end with the suffix: "TC". Objects all starts with the prefix: "jm"
 4544 followed by the group name. Attributes are identified with enums, and
 4545 so start with any lower case letter and have no special prefix.

| | | |
|------|------------------------------------|----|
| 4546 | | |
| 4547 | colorantConsumed | 67 |
| 4548 | colorantRequested | 66 |
| 4549 | deviceNameRequested | 56 |
| 4550 | documentCopiesCompleted | 61 |
| 4551 | documentCopiesRequested | 61 |
| 4552 | documentFormat | 58 |
| 4553 | documentFormatIndex | 57 |
| 4554 | documentName | 57 |
| 4555 | fileName | 57 |
| 4556 | finishing | 60 |
| 4557 | fullColorImpressionsCompleted | 63 |
| 4558 | highlightColorImpressionsCompleted | 64 |
| 4559 | impressionsCompletedCurrentCopy | 63 |
| 4560 | impressionsInterpreted | 63 |
| 4561 | impressionsSentToDevice | 63 |
| 4562 | impressionsSpooled | 63 |
| 4563 | jmAttributeInstanceIndex | 97 |
| 4564 | jmAttributeTypeIndex | 97 |
| 4565 | JmAttributeTypeTC | 50 |
| 4566 | jmAttributeValueAsInteger | 98 |
| 4567 | jmAttributeValueAsOctets | 99 |
| 4568 | JmBooleanTC | 41 |
| 4569 | JmFinishingTC | 39 |
| 4570 | jmGeneralAttributePersistence | 84 |
| 4571 | jmGeneralJobPersistence | 84 |
| 4572 | jmGeneralJobSetIndex | 82 |
| 4573 | jmGeneralJobSetName | 85 |
| 4574 | jmGeneralNewestActiveJobIndex | 83 |
| 4575 | jmGeneralNumberOfActiveJobs | 82 |
| 4576 | jmGeneralOldestActiveJobIndex | 83 |
| 4577 | jmJobIDJobIndex | 88 |
| 4578 | jmJobIDJobSetIndex | 88 |
| 4579 | jmJobImpressionsCompleted | 93 |
| 4580 | jmJobImpressionsPerCopyRequested | 93 |
| 4581 | jmJobIndex | 90 |
| 4582 | jmJobKOctetsPerCopyRequested | 92 |
| 4583 | jmJobKOctetsProcessed | 92 |
| 4584 | jmJobOwner | 94 |
| 4585 | JmJobServiceTypesTC | 71 |
| 4586 | JmJobSourcePlatformTypeTC | 38 |
| 4587 | jmJobState | 90 |
| 4588 | jmJobStateReasons1 | 91 |
| 4589 | JmJobStateReasons1TC | 72 |
| 4590 | JmJobStateReasons2TC | 76 |

| | | |
|------|----------------------------|----|
| 4591 | JmJobStateReasons3TC | 80 |
| 4592 | JmJobStateReasons4TC | 80 |
| 4593 | JmJobStateTC | 47 |
| 4594 | JmJobStringTC | 37 |
| 4595 | jmJobSubmissionID | 87 |
| 4596 | JmJobSubmissionIDTypeTC | 43 |
| 4597 | JmMediumTypeTC | 41 |
| 4598 | JmNaturalLanguageTagTC | 37 |
| 4599 | jmNumberOfInterveningJobs | 91 |
| 4600 | JmPrinterResolutionTC | 40 |
| 4601 | JmPrintQualityTC | 40 |
| 4602 | JmTimeStampTC | 38 |
| 4603 | JmTonerEconomyTC | 41 |
| 4604 | JmUTF8StringTC | 37 |
| 4605 | jobAccountName | 53 |
| 4606 | jobCodedCharSet | 52 |
| 4607 | jobCollationType | 62 |
| 4608 | jobComment | 57 |
| 4609 | jobCompletionTime | 68 |
| 4610 | jobCopiesCompleted | 61 |
| 4611 | jobCopiesRequested | 61 |
| 4612 | jobHold | 59 |
| 4613 | jobHoldUntil | 59 |
| 4614 | jobKOctetsTransferred | 62 |
| 4615 | jobName | 54 |
| 4616 | jobNaturalLanguageTag | 53 |
| 4617 | jobOriginatingHost | 56 |
| 4618 | jobPriority | 58 |
| 4619 | jobProcessAfterDateAndTime | 59 |
| 4620 | jobProcessingCPUtime | 68 |
| 4621 | jobServiceTypes | 55 |
| 4622 | jobSourceChannelIndex | 55 |
| 4623 | jobSourcePlatformType | 55 |
| 4624 | jobStartedBeingHeldTime | 68 |
| 4625 | jobStartedProcessingTime | 68 |
| 4626 | jobStateReasons2 | 51 |
| 4627 | jobStateReasons3 | 51 |
| 4628 | jobStateReasons4 | 51 |
| 4629 | jobSubmissionTime | 67 |
| 4630 | jobSubmissionToServerTime | 67 |
| 4631 | jobURI | 53 |
| 4632 | mediumConsumed | 66 |
| 4633 | mediumRequested | 66 |
| 4634 | numberOfDocuments | 56 |
| 4635 | other | 50 |
| 4636 | outputBin | 59 |
| 4637 | pagesCompleted | 64 |
| 4638 | pagesCompletedCurrentCopy | 65 |
| 4639 | pagesRequested | 64 |
| 4640 | physicalDevice | 56 |
| 4641 | printerResolutionRequested | 60 |
| 4642 | printerResolutionUsed | 60 |
| 4643 | printQualityRequested | 60 |

| | | |
|------|---------------------------------|----|
| 4644 | printQualityUsed | 60 |
| 4645 | processingMessage | 51 |
| 4646 | processingMessageNaturalLangTag | 52 |
| 4647 | queueNameRequested | 56 |
| 4648 | serverAssignedJobName | 54 |
| 4649 | sheetCompletedCopyNumber | 62 |
| 4650 | sheetCompletedDocumentNumber | 62 |
| 4651 | sheetsCompleted | 65 |
| 4652 | sheetsCompletedCurrentCopy | 65 |
| 4653 | sheetsRequested | 65 |
| 4654 | sides | 60 |
| 4655 | submittingApplicationName | 55 |
| 4656 | submittingServerName | 55 |
| 4657 | tonerDensityRequested | 60 |
| 4658 | tonerDensityUsed | 61 |
| 4659 | tonerEcomonyRequested | 60 |
| 4660 | tonerEcomonyUsed | 60 |
| 4661 | | |