

1 INTERNET-DRAFT

R. Bergman  
Dataproducts Corp.  
T. Hastings  
Xerox Corporation  
S. Isaacson  
Novell, Inc.  
H. Lewis  
IBM Corp.  
October 2, 1998

10 Job Monitoring MIB - V1.2  
11 <draft-ietf-printmib-job-monitor-08.txt>

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 April 2, 1998.

29  
30

#### Abstract

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



45		
46		TABLE OF CONTENTS
47	1.	INTRODUCTION <span style="float: right;">6</span>
48	1.1	Types of Information in the MIB <span style="float: right;">6</span>
49	1.2	Types of Job Monitoring Applications <span style="float: right;">8</span>
50	2.	TERMINOLOGY AND JOB MODEL <span style="float: right;">9</span>
51	2.1	System Configurations for the Job Monitoring MIB <span style="float: right;">12</span>
52	2.1.1	Configuration 1 - client-printer <span style="float: right;">12</span>
53	2.1.2	Configuration 2 - client-server-printer - agent in the
54		server <span style="float: right;">13</span>
55	2.1.3	Configuration 3 - client-server-printer - client monitors
56		printer agent and server <span style="float: right;">14</span>
57	3.	MANAGED OBJECT USAGE <span style="float: right;">16</span>
58	3.1	Conformance Considerations <span style="float: right;">16</span>
59	3.1.1	Conformance Terminology <span style="float: right;">16</span>
60	3.1.2	Agent Conformance Requirements <span style="float: right;">16</span>
61	3.1.2.1	MIB II System Group objects <span style="float: right;">17</span>
62	3.1.2.2	MIB II Interface Group objects <span style="float: right;">17</span>
63	3.1.2.3	Printer MIB objects <span style="float: right;">17</span>
64	3.1.3	Job Monitoring Application Conformance Requirements <span style="float: right;">17</span>
65	3.2	The Job Tables and the Oldest Active and Newest Active Indexes <span style="float: right;">18</span>
66	3.3	The Attribute Mechanism <span style="float: right;">20</span>
67	3.3.1	Conformance of Attribute Implementation <span style="float: right;">20</span>
68	3.3.2	Useful, 'Unknown', and 'Other' Values for Objects and
69		Attributes <span style="float: right;">21</span>
70	3.3.3	Index Value Attributes <span style="float: right;">21</span>
71	3.3.4	Data Sub-types and Attribute Naming Conventions <span style="float: right;">22</span>
72	3.3.5	Single-Value (Row) Versus Multi-Value (MULTI-ROW)
73		Attributes <span style="float: right;">23</span>
74	3.3.6	Requested Objects and Attributes <span style="float: right;">23</span>
75	3.3.7	Consumption Attributes <span style="float: right;">23</span>
76	3.3.8	Attribute Specifications <span style="float: right;">24</span>
77	3.4	Monitoring Job Progress <span style="float: right;">44</span>
78	3.5	Job Identification <span style="float: right;">48</span>
79	3.6	Internationalization Considerations <span style="float: right;">49</span>
80	3.6.1	Text generated by the server or device <span style="float: right;">49</span>
81	3.6.2	Text supplied by the job submitter <span style="float: right;">50</span>
82	3.6.3	'DateAndTime' for representing the date and time <span style="float: right;">51</span>

83	3.7 IANA and PWG Registration Considerations	51
84	3.7.1 PWG Registration of enums	52
85	3.7.1.1 Type 1 enumerations	<b>52</b>
86	3.7.1.2 Type 2 enumerations	<b>52</b>
87	3.7.1.3 Type 3 enumeration	<b>53</b>
88	3.7.2 PWG Registration of type 2 bit values	53
89	3.7.3 PWG Registration of Job Submission Id Formats	53
90	3.7.4 PWG Registration of MIME types/sub-types for document-	
91	formats 53	
92	3.8 Security Considerations	54
93	3.8.1 Read-Write objects	54
94	3.8.2 Read-Only Objects In Other User's Jobs	54
95	3.9 Notifications	54
96	4. MIB SPECIFICATION	<b>54</b>
97	Textual conventions for this MIB module	56
98	JmUTF8StringTC	56
99	JmJobStringTC	56
100	JmNaturalLanguageTagTC	56
101	JmTimeStampTC	56
102	JmJobSourcePlatformTypeTC	57
103	JmFinishingTC	58
104	JmPrintQualityTC	59
105	JmPrinterResolutionTC	59
106	JmTonerEconomyTC	60
107	JmBooleanTC	60
108	JmMediumTypeTC	60
109	JmJobCollationTypeTC	61
110	JmJobSubmissionIDTypeTC	62
111	JmJobStateTC	67
112	JmAttributeTypeTC	70
113	JmJobServiceTypesTC	73
114	JmJobStateReasons1TC	75
115	JmJobStateReasons2TC	79
116	JmJobStateReasons3TC	83
117	JmJobStateReasons4TC	83
118	The General Group (MANDATORY)	84
119	jmGeneralJobSetIndex (Int32(1..32767))	85
120	jmGeneralNumberOfActiveJobs (Int32(0..))	85
121	jmGeneralOldestActiveJobIndex (Int32(0..))	86
122	jmGeneralNewestActiveJobIndex (Int32(0..))	86
123	jmGeneralJobPersistence (Int32(15..))	87
124	jmGeneralAttributePersistence (Int32(15..))	87
125	jmGeneralJobSetName (UTF8String63)	88
126	The Job ID Group (MANDATORY)	88
127	jmJobSubmissionID (OCTET STRING(SIZE(48)))	90

128	jmJobIDJobSetIndex (Int32(0..32767))	91
129	jmJobIDJobIndex (Int32(0..))	91
130	The Job Group (MANDATORY)	91
131	jmJobIndex (Int32(1..))	93
132	jmJobState (JmJobStateTC)	93
133	jmJobStateReasons1 (JmJobStateReasons1TC)	94
134	jmNumberOfInterveningJobs (Int32(-2..))	94
135	jmJobKOctetsPerCopyRequested (Int32(-2..))	95
136	jmJobKOctetsProcessed (Int32(-2..))	95
137	jmJobImpressionsPerCopyRequested (Int32(-2..))	96
138	jmJobImpressionsCompleted (Int32(-2..))	96
139	jmJobOwner (JobString63)	97
140	The Attribute Group (MANDATORY)	97
141	jmAttributeTypeIndex (JmAttributeTypeTC)	100
142	jmAttributeInstanceIndex (Int32(1..32767))	100
143	jmAttributeValueAsInteger (Int32(-2..))	101
144	jmAttributeValueAsOctets (Octets63)	102
145	5. APPENDIX A - IMPLEMENTING THE JOB LIFE CYCLE	<b>105</b>
146	6. APPENDIX B - SUPPORT OF JOB SUBMISSION PROTOCOLS	<b>106</b>
147	7. REFERENCES	<b>106</b>
148	8. AUTHOR'S ADDRESSES	<b>108</b>
149	9. CHANGE HISTORY	<b>110</b>
150	9.1 Changes to produce version 1.1, dated October 1, 1998	110
151	9.2 Changes to produce version 1.2, dated October 2, 1998	111
152	10. INDEX	<b>112</b>
153		

## 154 Job Monitoring MIB

## 155 1 Introduction

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

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

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

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

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

194 User:

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

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

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

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

205 Operator:

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

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

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

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

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

218 Capacity Planner:

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

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

223 Accountant:

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

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

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

## 235 1.2 Types of Job Monitoring Applications

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

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

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



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

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

## 281 2 Terminology and Job Model

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

380 Monitor: Short for Job Monitoring Application.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

417 **2.1 System Configurations for the Job Monitoring MIB**

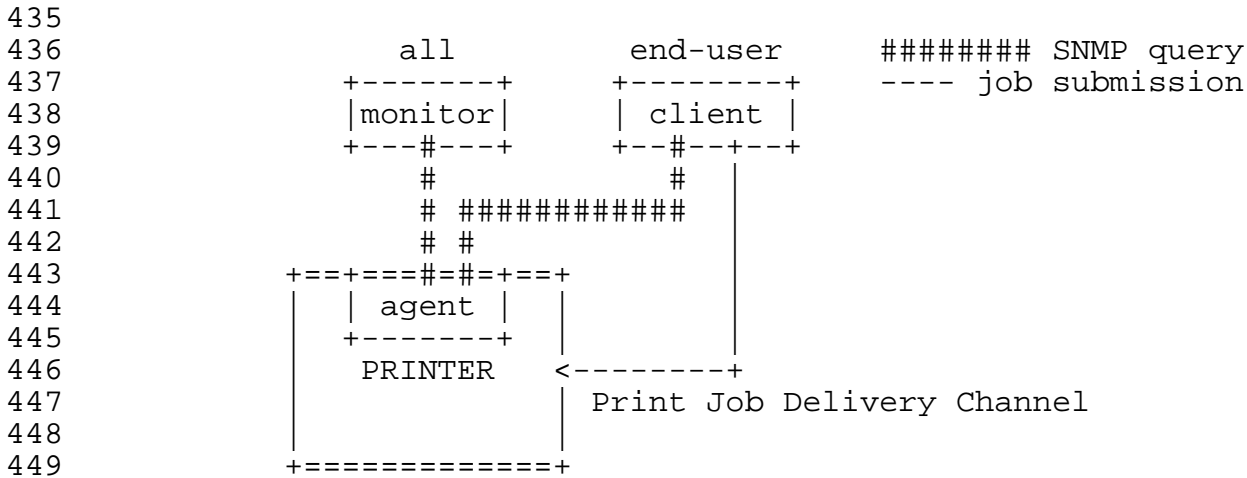
418 This section enumerates the three configurations in which the Job  
419 Monitoring MIB is intended to be used. To simplify the pictures, the  
420 *devices* are shown as *printers*. See section 1.1 entitled "Types of  
421 Information in the MIB".

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

425 2.1.1 Configuration 1 - client-printer

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

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



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

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

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

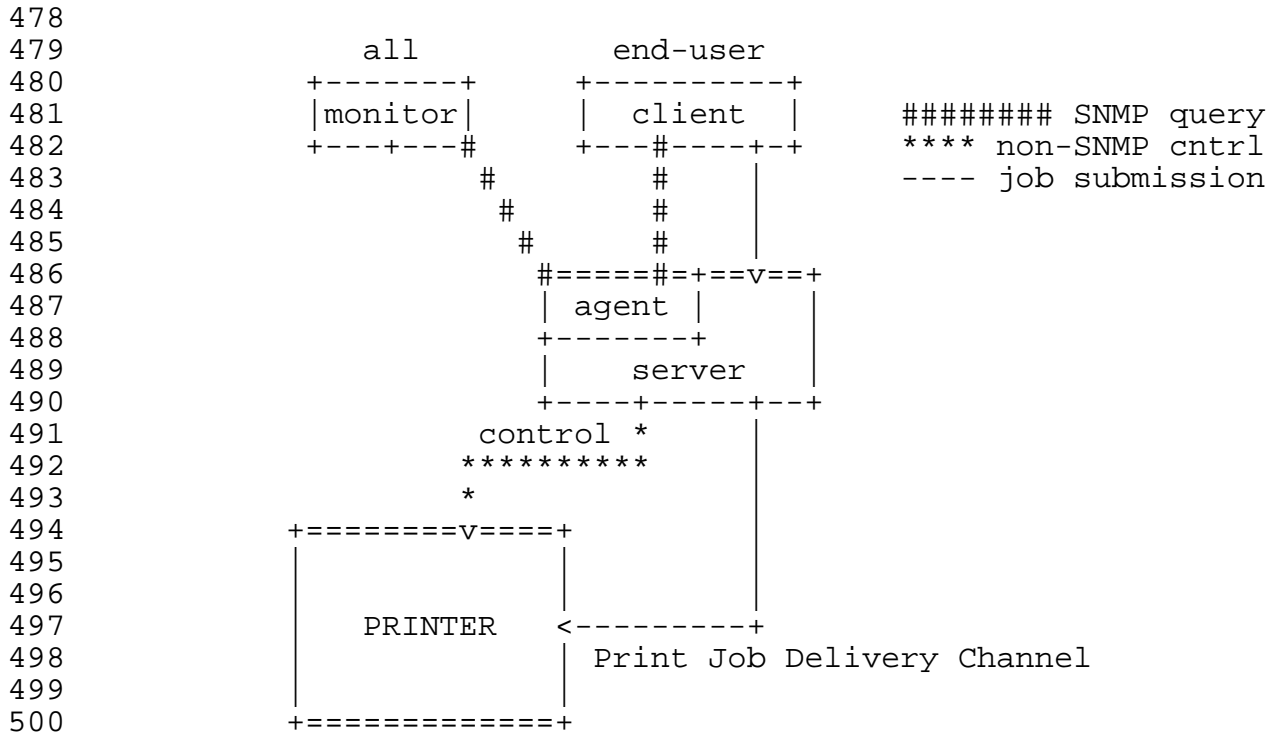
458 2.1.2 Configuration 2 - client-server-printer - agent in the server

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

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

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

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



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

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

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

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

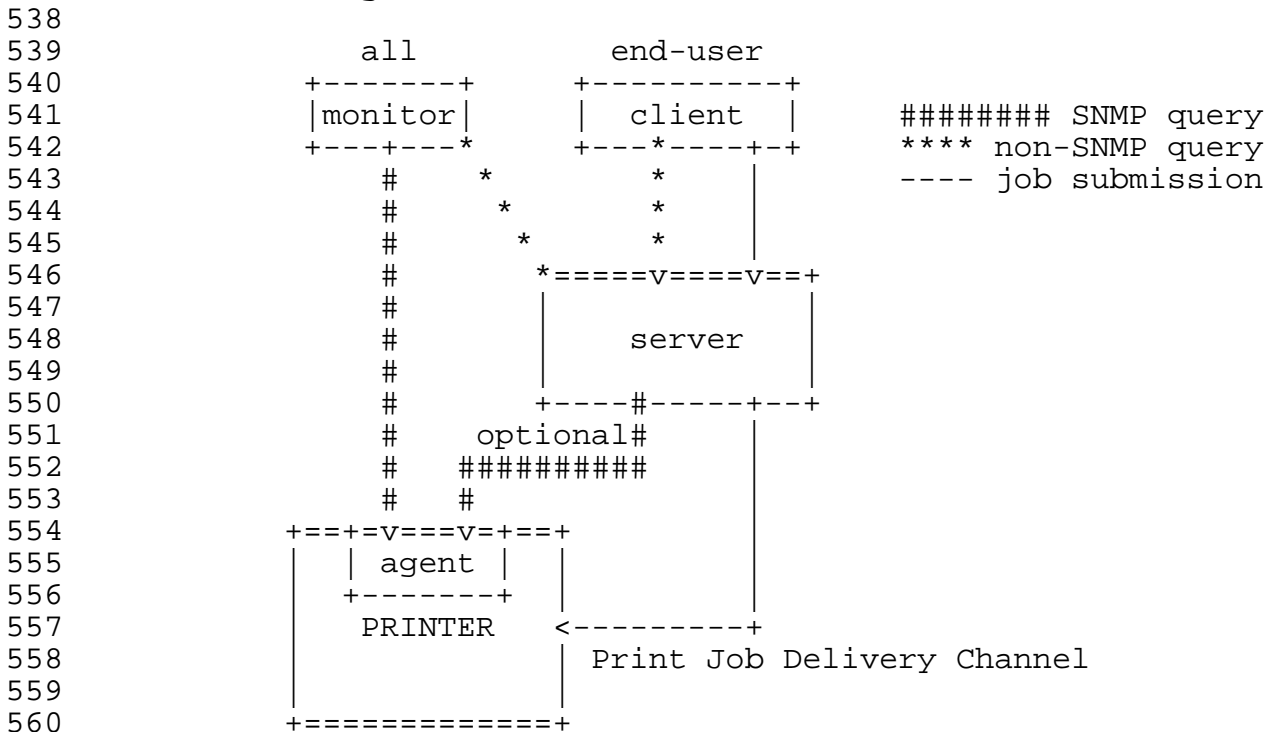
514 In the client-server-printer configuration 3, the client(s) submit jobs  
 515 to an intermediate server by some network connection, *not* directly to  
 516 the printer. That server does *not* contain a Job Monitoring MIB agent.

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

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

523 In such configurations, the server deletes its copy of the job  
 524 from the server after submitting the job to the printer usually  
 525 almost immediately (before the job does much processing, if  
 526 any).

527 In configuration 3, the agent (in the printer) SHALL keep the values of  
 528 the objects in the Job Monitoring MIB that the agent implements updated  
 529 for a job that the server has submitted to the printer. The agent  
 530 SHALL obtain information about the jobs submitted to the printer from  
 531 the server (either in the job submission protocol, in the document  
 532 data, or by direct query of the server), in order to populate some of  
 533 the objects the Job Monitoring MIB in the printer. The agent in the  
 534 printer SHALL keep the job in the Job Monitoring MIB as long as the job  
 535 is in the Printer, and longer in order to implement the completed state  
 536 in which monitoring programs can copy out the accounting data from the  
 537 Job Monitoring MIB.



561 Figure 2-3 - Configuration 3 - client-server-printer - client monitors  
 562 printer agent and server

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

- 565 1. Multiple clients MAY submit jobs to a server.
- 566 2. Multiple clients MAY monitor a server.
- 567 3. Multiple monitors MAY monitor a server.
- 568 4. A client MAY submit jobs to multiple servers.
- 569 5. A monitor MAY monitor multiple servers.
- 570 6. Multiple servers MAY submit jobs to a printer.
- 571 7. Multiple servers MAY control a printer.

## 572 3 Managed Object Usage

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

574 **3.1 Conformance Considerations**

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

## 578 3.1.1 Conformance Terminology

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

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

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

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

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

## 594 3.1.2 Agent Conformance Requirements

595 A conforming agent:

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

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



## 608 3.1.2.1 MIB II System Group objects

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

## 612 3.1.2.2 MIB II Interface Group objects

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

## 616 3.1.2.3 Printer MIB objects

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

## 624 3.1.3 Job Monitoring Application Conformance Requirements

625 A conforming job monitoring application:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

### 727 3.3 The Attribute Mechanism

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

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

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

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

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

#### 758 3.3.1 Conformance of Attribute Implementation

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

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

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

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

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

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

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

## 799 3.3.3 Index Value Attributes

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

## 808 3.3.4 Data Sub-types and Attribute Naming Conventions

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

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

'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.

829

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

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

841 NOTE - Duplicates are allowed for 'extensive' 'MULTI-ROW' attributes,  
842 such as fileName(34) or documentName(35) which are specified to be  
843 'per-document' attributes, but are *not* allowed for 'intensive' 'MULTI-  
844 ROW' attributes, such as mediumConsumed(171) and documentFormat(38)  
845 which are specified to be 'per-job' attributes.

## 846 3.3.6 Requested Objects and Attributes

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

## 857 3.3.7 Consumption Attributes

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

## 866 3.3.8 Attribute Specifications

867 This section specifies the job attributes.

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

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

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

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

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

889 NOTE: No attribute name exceeds 31 characters.

890 The standard attribute types are:

891	jmAttributeTypeIndex	Datatype
892	-----	-----
893		
894		
895	other(1),	Integer32 (-2..2147483647)
896		AND/OR
897		OCTET STRING(SIZE(0..63))
898	INTEGER: and/or OCTETS:	An attribute that is not in the
899	list and/or that has not been	approved and registered with
900	the PWG.	



```
901 ++++++
902 + Job State attributes
903 +
904 + The following attributes specify the state of a job.
905 ++++++
906
907 jobStateReasons2(3),                JmJobStateReasons2TC
908     INTEGER: Additional information about the job's current
909     state that augments the jmJobState object. See the
910     description under the JmJobStateReasons1TC textual-
911     convention.
912
913 jobStateReasons3(4),                JmJobStateReasons3TC
914     INTEGER: Additional information about the job's current
915     state that augments the jmJobState object. See the
916     description under JmJobStateReasons1TC textual-convention.
917
918 jobStateReasons4(5),                JmJobStateReasons4TC
919     INTEGER: Additional information about the job's current
920     state that augments the jmJobState object. See the
921     description under JmJobStateReasons1TC textual-convention.
922
923 processingMessage(6),                JmUTF8StringTC (SIZE(0..63))
924     OCTETS: MULTI-ROW: A coded character set message that is
925     generated by the server or device during the processing of
926     the job as a simple form of processing log to show progress
927     and any problems. The natural language of each value is
928     specified by the corresponding
929     processingMessageNaturalLangTag(7) value.
930
931     NOTE - This attribute is intended for such conditions as
932     interpreter messages, rather than being the printable form
933     of the jmJobState and jmJobStateReasons1 objects and
934     jobStateReasons2, jobStateReasons3, and jobStateReasons4
935     attributes. In order to produce a localized printable form
936     of these job state objects/attribute, a management
937     application SHOULD produce a message from their enum and
938     bit values.
939
940     NOTE - There is no job description attribute in IPP/1.0
941     that corresponds to this attribute and this attribute does
942     not correspond to the IPP/1.0 'job-state-message' job
943     description attribute, which is just a printable form of
944     the IPP 'job-state' and 'job-state-reasons' job attributes.
945
946     There is no restriction for the same message occurring in
947     multiple rows.
```

948  
949 processingMessageNaturalLangTag(7), OCTET STRING(SIZE(0..63))  
950 OCTETS: MULTI-ROW: The natural language of the  
951 corresponding processingMessage(6) attribute value. See  
952 section 3.6.1, entitled 'Text generated by the server or  
953 device'.  
954  
955 If the agent does not know the natural language of the job  
956 processing message, the agent SHALL either (1) return a  
957 zero length string value for the  
958 processingMessageNaturalLangTag(7) attribute or (2) not  
959 return the processingMessageNaturalLangTag(7) attribute for  
960 the job.  
961  
962 There is no restriction for the same tag occurring in  
963 multiple rows, since when this attribute is implemented, it  
964 SHOULD have a value row for each corresponding  
965 processingMessage(6) attribute value row.  
966  
967 jobCodedCharSet(8), CodedCharSet  
968 INTEGER: The MIBenum identifier of the coded character set  
969 that the agent is using to represent coded character set  
970 objects and attributes of type 'JmJobStringTC'. These  
971 coded character set objects and attributes are either: (1)  
972 supplied by the job submitting client or (2) defaulted by  
973 the server or device when omitted by the job submitting  
974 client. The agent SHALL represent these objects and  
975 attributes in the MIB either (1) in the coded character set  
976 as they were submitted or (2) MAY convert the coded  
977 character set to another coded character set or encoding  
978 scheme as identified by the jobCodedCharSet(8) attribute.  
979 See section 3.6.2, entitled 'Text supplied by the job  
980 submitter'.  
981  
982 These MIBenum values are assigned by IANA [IANA-charsets]  
983 when the coded character sets are registered. The coded  
984 character set SHALL be one of the ones registered with IANA  
985 [IANA] and the enum value uses the CodedCharSet textual-  
986 convention from the Printer MIB. See the JmJobStringTC  
987 textual-convention.  
988  
989 If the agent does not know what coded character set was  
990 used by the job submitting client, the agent SHALL either  
991 (1) return the 'unknown(2)' value for the  
992 jobCodedCharSet(8) attribute or (2) not return the  
993 jobCodedCharSet(8) attribute for the job.

994           jobNaturalLanguageTag(9),                   OCTET STRING(SIZE(0..63))  
995            OCTETS: The natural language of the job attributes supplied  
996            by the job submitter or defaulted by the server or device  
997            for the job, i.e., all objects and attributes represented  
998            by the 'JmJobStringTC' textual-convention, such as jobName,  
999            mediumRequested, etc. See Section 3.6.2, entitled 'Text  
1000            supplied by the job submitter'.  
1001  
1002            If the agent does not know what natural language was used  
1003            by the job submitting client, the agent SHALL either (1)  
1004            return a zero length string value for the  
1005            jobNaturalLanguageTag(9) attribute or (2) not return  
1006            jobNaturalLanguageTag(9) attribute for the job.  
1007  
1008            +++++  
1009            + Job Identification attributes  
1010            +  
1011            + The following attributes help an end user, a system  
1012            + operator, or an accounting program identify a job.  
1013            +++++  
1014  
1015            jobURI(20),                                OCTET STRING(SIZE(0..63))  
1016            OCTETS: MULTI-ROW: The job's Universal Resource  
1017            Identifier (URI) [RFC-1738]. See IPP [ipp-model] for  
1018            example usage.  
1019  
1020            NOTE - The agent may be able to generate this value on each  
1021            SNMP Get operation from smaller values, rather than having  
1022            to store the entire URI.  
1023  
1024            If the URI exceeds 63 octets, the agent SHALL use multiple  
1025            values, with the next 63 octets coming in the second value,  
1026            etc.  
1027  
1028            NOTE - IPP [ipp-model] has a 1023-octet maximum length for  
1029            a URI, though the URI standard itself and HTTP/1.1 specify  
1030            no maximum length.  
1031  
1032            jobAccountName(21),                        OCTET STRING(SIZE(0..63))  
1033            OCTETS: Arbitrary binary information which MAY be coded  
1034            character set data or encrypted data supplied by the  
1035            submitting user for use by accounting services to allocate  
1036            or categorize charges for services provided, such as a  
1037            customer account name or number.  
1038  
1039            NOTE: This attribute NEED NOT be printable characters.  
1040

1041 serverAssignedJobName(22), JmJobStringTC (SIZE(0..63))  
1042 OCTETS: Configuration 3 only: The human readable string  
1043 name, number, or ID of the job as assigned by the server  
1044 that submitted the job to the device that the agent is  
1045 providing access to with this MIB.  
1046  
1047 NOTE - This attribute is intended for enabling a user to  
1048 find his/her job that a server submitted to a device when  
1049 either the client does not support the jmJobSubmissionID or  
1050 the server does not pass the jmJobSubmissionID through to  
1051 the device.  
1052

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

1087           jobServiceTypes(24),                           JmJobServiceTypesTC  
1088            INTEGER: Specifies the type(s) of service to which the job  
1089            has been submitted (print, fax, scan, etc.). The service  
1090            type is bit encoded with each job service type so that more  
1091            general and arbitrary services can be created, such as  
1092            services with more than one destination type, or ones with  
1093            only a source or only a destination. For example, a job  
1094            service might scan, faxOut, and print a single job. In  
1095            this case, three bits would be set in the jobServiceTypes  
1096            attribute, corresponding to the hexadecimal values: 0x8 +  
1097            0x20 + 0x4, respectively, yielding: 0x2C.  
1098  
1099            Whether this attribute is set from a job attribute supplied  
1100            by the job submission client or is set by the recipient job  
1101            submission server or device depends on the job submission  
1102            protocol. This attribute SHALL be implemented if the  
1103            server or device has other types in addition to or instead  
1104            of printing.  
1105  
1106            One of the purposes of this attribute is to permit a  
1107            requester to filter out jobs that are not of interest. For  
1108            example, a printer operator may only be interested in jobs  
1109            that include printing.  
1110  
1111            jobSourceChannelIndex(25),                   Integer32 (0..2147483647)  
1112            INTEGER: The index of the row in the associated Printer  
1113            MIB[print-mib] of the channel which is the source of the  
1114            print job.  
1115  
1116            jobSourcePlatformType(26),                   JmJobSourcePlatformTypeTC  
1117            INTEGER: The source platform type of the immediate  
1118            upstream submitter that submitted the job to the server  
1119            (configuration 2) or device (configuration 1 and 3) to  
1120            which the agent is providing access. For configuration 1,  
1121            this is the type of the client that submitted the job to  
1122            the device; for configuration 2, this is the type of the  
1123            client that submitted the job to the server; and for  
1124            configuration 3, this is the type of the server that  
1125            submitted the job to the device.  
1126  
1127            submittingServerName(27),                    JmJobStringTC (SIZE(0..63))  
1128            OCTETS: For configuration 3 only: The administrative name  
1129            of the server that submitted the job to the device.  
1130  
1131            submittingApplicationName(28),               JmJobStringTC (SIZE(0..63))  
1132            OCTETS: The name of the client application (not the server  
1133            in configuration 3) that submitted the job to the server or  
1134            device.  
1135

1136           jobOriginatingHost(29),                   JmJobStringTC (SIZE(0..63))  
1137            OCTETS: The name of the client host (not the server host  
1138            name in configuration 3) that submitted the job to the  
1139            server or device.  
1140

1141           deviceNameRequested(30),                   JmJobStringTC (SIZE(0..63))  
1142            OCTETS: The administratively defined coded character set  
1143            name of the target device requested by the submitting user.  
1144            For configuration 1, its value corresponds to the Printer  
1145            MIB[print-mib]: prtGeneralPrinterName object. For  
1146            configuration 2 and 3, its value is the name of the logical  
1147            or physical device that the user supplied to indicate to  
1148            the server on which device(s) they wanted the job to be  
1149            processed.  
1150

1151           queueNameRequested(31),                    JmJobStringTC (SIZE(0..63))  
1152            OCTETS: The administratively defined coded character set  
1153            name of the target queue requested by the submitting user.  
1154            For configuration 1, its value corresponds to the queue in  
1155            the device for which the agent is providing access. For  
1156            configuration 2 and 3, its value is the name of the queue  
1157            that the user supplied to indicate to the server on which  
1158            device(s) they wanted the job to be processed.  
1159

1160            NOTE - typically an implementation SHOULD support either  
1161            the deviceNameRequested or queueNameRequested attribute,  
1162            but not both.  
1163

1164           physicalDevice(32),                        hrDeviceIndex  
1165    AND/OR  
1166    JmUTF8StringTC (SIZE(0..63))  
1167            INTEGER: MULTI-ROW: The index of the physical device MIB  
1168            instance requested/used, such as the Printer MIB[print-  
1169            mib]. This value is an hrDeviceIndex value. See the Host  
1170            Resources MIB[hr-mib].  
1171

1172            AND/OR

1173

1174            OCTETS: MULTI-ROW: The name of the physical device to  
1175            which the job is assigned.  
1176

1177            numberOfDocuments(33),                   Integer32 (-2..2147483647)  
1178            INTEGER: The number of documents in this job.  
1179

1180            The agent SHOULD return this attribute if the job has more  
1181            than one document.  
1182



```

1224     documentFormat(38),                               PrtInterpreterLangFamilyTC
1225                                                     AND/OR
1226                                                     OCTET STRING(SIZE(0..63))
1227     INTEGER: MULTI-ROW: The interpreter language family
1228     corresponding to the Printer MIB[print-mib]
1229     prtInterpreterLangFamily object, that this job
1230     requires/uses. A document or a job MAY use more than one
1231     PDL or control language.
1232
1233     AND/OR
1234
1235     OCTETS: MULTI-ROW: The document format registered as a
1236     media type[iana-media-types], i.e., the name of the MIME
1237     content-type/subtype. Examples: 'application/postscript',
1238     'application/vnd.hp-PCL', 'application/pdf', 'text/plain'
1239     (US-ASCII SHALL be assumed), 'text/plain; charset=iso-8859-
1240     1', and 'application/octet-stream'. The IPP 'document-
1241     format' job attribute uses these same values with the same
1242     semantics. See the IPP [ipp-model] 'mimeMediaType'
1243     attribute syntax and the document-format attribute for
1244     further examples and explanation.
1245
1246     ++++++
1247     + Job Parameter attributes
1248     +
1249     + The following attributes represent input parameters
1250     + supplied by the submitting client in the job submission
1251     + protocol.
1252     ++++++
1253
1254     jobPriority(50),                                     Integer32 (-2..100)
1255     INTEGER: The priority for scheduling the job. It is used
1256     by servers and devices that employ a priority-based
1257     scheduling algorithm.
1258
1259     A higher value specifies a higher priority. The value 1 is
1260     defined to indicate the lowest possible priority (a job
1261     which a priority-based scheduling algorithm SHALL pass over
1262     in favor of higher priority jobs). The value 100 is
1263     defined to indicate the highest possible priority.
1264     Priority is expected to be evenly or 'normally' distributed
1265     across this range. The mapping of vendor-defined priority
1266     over this range is implementation-specific. -2 indicates
1267     unknown.
1268

```



1269           jobProcessAfterDateAndTime(51),     DateAndTime (SNMPv2-TC)  
1270           OCTETS: The calendar date and time of day after which the  
1271           job SHALL become a candidate to be scheduled for  
1272           processing. If the value of this attribute is in the  
1273           future, the server SHALL set the value of the job's  
1274           jmJobState object to pendingHeld and add the  
1275           jobProcessAfterSpecified bit value to the job's  
1276           jmJobStateReasons1 object. When the specified date and  
1277           time arrives, the server SHALL remove the  
1278           jobProcessAfterSpecified bit value from the job's  
1279           jmJobStateReasons1 object and, if no other reasons remain,  
1280           SHALL change the job's jmJobState object to pending.  
1281  
1282           jobHold(52),                             JmBooleanTC  
1283           INTEGER: If the value is 'true(4)', a client has  
1284           explicitly specified that the job is to be held until  
1285           explicitly released. Until the job is explicitly released  
1286           by a client, the job SHALL be in the pendingHeld state with  
1287           the jobHoldSpecified value in the jmJobStateReasons1  
1288           attribute.  
1289  
1290           jobHoldUntil(53),                        JmJobStringTC (SIZE(0..63))  
1291           OCTETS: The named time period during which the job SHALL  
1292           become a candidate for processing, such as 'evening',  
1293           'night', 'weekend', 'second-shift', 'third-shift', etc.,  
1294           (supported values configured by the system administrator).  
1295           See IPP [ipp-model] for the standard keyword values. Until  
1296           that time period arrives, the job SHALL be in the  
1297           pendingHeld state with the jobHoldUntilSpecified value in  
1298           the jmJobStateReasons1 object. The value 'no-hold' SHALL  
1299           indicate explicitly that no time period has been specified;  
1300           the absence of this attribute SHALL indicate implicitly  
1301           that no time period has been specified.  
1302  
1303           outputBin(54),                            Integer32 (0..2147483647)  
1304   AND/OR  
1305   JmJobStringTC (SIZE(0..63))  
1306           INTEGER: MULTI-ROW: The output subunit index in the  
1307           Printer MIB[print-mib]  
1308  
1309           AND/OR  
1310  
1311           OCTETS: MULTI-ROW: the name or number (represented as  
1312           ASCII digits) of the output bin to which all or part of the  
1313           job is placed in.  
1314

```
1315 sides(55), Integer32 (-2..2)
1316     INTEGER: MULTI-ROW: The number of sides, '1' or '2', that
1317     any document in this job requires/used.
1318
1319 finishing(56), JmFinishingTC
1320     INTEGER: MULTI-ROW: Type of finishing that any document
1321     in this job requires/used.
1322
1323
1324 ++++++
1325 + Image Quality attributes (requested and consumed)
1326 +
1327 + For devices that can vary the image quality.
1328 ++++++
1329
1330 printQualityRequested(70), JmPrintQualityTC
1331     INTEGER: MULTI-ROW: The print quality selection requested
1332     for a document in the job for printers that allow quality
1333     differentiation.
1334
1335 printQualityUsed(71), JmPrintQualityTC
1336     INTEGER: MULTI-ROW: The print quality selection actually
1337     used by a document in the job for printers that allow
1338     quality differentiation.
1339
1340 printerResolutionRequested(72), JmPrinterResolutionTC
1341     OCTETS: MULTI-ROW: The printer resolution requested for a
1342     document in the job for printers that support resolution
1343     selection.
1344
1345 printerResolutionUsed(73), JmPrinterResolutionTC
1346     OCTETS: MULTI-ROW: The printer resolution actually used
1347     by a document in the job for printers that support
1348     resolution selection.
1349
1350 tonerEcomonyRequested(74), JmTonerEcomonyTC
1351     INTEGER: MULTI-ROW: The toner economy selection requested
1352     for documents in the job for printers that allow toner
1353     economy differentiation.
1354
1355 tonerEcomonyUsed(75), JmTonerEcomonyTC
1356     INTEGER: MULTI-ROW: The toner economy selection actually
1357     used by documents in the job for printers that allow toner
1358     economy differentiation.
1359
1360 tonerDensityRequested(76) Integer32 (-2..100)
1361     INTEGER: MULTI-ROW: The toner density requested for a
1362     document in this job for devices that can vary toner
1363     density levels. Level 1 is the lowest density and level
1364     100 is the highest density level. Devices with a smaller
1365     range, SHALL map the 1-100 range evenly onto the
1366     implemented range.
```

1367  
1368 tonerDensityUsed(77), Integer32 (-2..100)  
1369 INTEGER: MULTI-ROW: The toner density used by documents  
1370 in this job for devices that can vary toner density levels.  
1371 Level 1 is the lowest density and level 100 is the highest  
1372 density level. Devices with a smaller range, SHALL map the  
1373 1-100 range evenly onto the implemented range.  
1374  
1375 ++++++  
1376 + Job Progress attributes (requested and consumed)  
1377 +  
1378 + Pairs of these attributes can be used by monitoring  
1379 + applications to show an indication of relative progress  
1380 + to users. See section 3.4, entitled '**Monitoring Job**  
1381 **Progress**'.  
1382 ++++++  
1383  
1384 jobCopiesRequested(90), Integer32 (-2..2147483647)  
1385 INTEGER: The number of copies of the entire job that are  
1386 to be produced.  
1387  
1388 jobCopiesCompleted(91), Integer32 (-2..2147483647)  
1389 INTEGER: The number of copies of the entire job that have  
1390 been completed so far.  
1391  
1392 documentCopiesRequested(92), Integer32 (-2..2147483647)  
1393 INTEGER: The total count of the number of document copies  
1394 requested for the job as a whole. If there are documents  
1395 A, B, and C, and document B is specified to produce 4  
1396 copies, the number of document copies requested is 6 for  
1397 the job.  
1398  
1399 This attribute SHALL be used only when a job has multiple  
1400 documents. The jobCopiesRequested attribute SHALL be used  
1401 when the job has only one document.  
1402  
1403 documentCopiesCompleted(93), Integer32 (-2..2147483647)  
1404 INTEGER: The total count of the number of document copies  
1405 completed so far for the job as a whole. If there are  
1406 documents A, B, and C, and document B is specified to  
1407 produce 4 copies, the number of document copies starts a 0  
1408 and runs up to 6 for the job as the job processes.  
1409  
1410 This attribute SHALL be used only when a job has multiple  
1411 documents. The jobCopiesCompleted attribute SHALL be used  
1412 when the job has only one document.  
1413

1414           jobKOctetsTransferred(94),           Integer32 (-2..2147483647)  
1415            INTEGER: The number of K (1024) octets transferred to the  
1416            server or device to which the agent is providing access.  
1417            This count is independent of the number of copies of the  
1418            job or documents that will be produced, but it is only a  
1419            measure of the number of bytes transferred to the server or  
1420            device.  
1421  
1422            The agent SHALL round the actual number of octets  
1423            transferred up to the next higher K. Thus 0 octets SHALL  
1424            be represented as '0', 1-1024 octets SHALL BE represented  
1425            as '1', 1025-2048 SHALL be '2', etc. When the job  
1426            completes, the values of the jmJobKOctetsPerCopyRequested  
1427            object and the jobKOctetsTransferred attribute SHALL be  
1428            equal.  
1429  
1430            NOTE - The jobKOctetsTransferred can be used with the  
1431            jmJobKOctetsPerCopyRequested object in order to produce a  
1432            relative indication of the progress of the job for agents  
1433            that do not implement the jmJobKOctetsProcessed object.  
1434  
1435            sheetCompletedCopyNumber(95),           Integer32 (-2..2147483647)  
1436            INTEGER: The number of the copy being stacked for the  
1437            current document. This number starts at 0, is set to 1  
1438            when the first sheet of the first copy for each document is  
1439            being stacked and is equal to n where n is the nth sheet  
1440            stacked in the current document copy. See section 3.4 ,  
1441            entitled 'Monitoring Job Progress'.  
1442  
1443            sheetCompletedDocumentNumber(96), Integer32 (-2..2147483647)  
1444            INTEGER: The ordinal number of the document in the job  
1445            that is currently being stacked. This number starts at 0,  
1446            increments to 1 when the first sheet of the first document  
1447            in the job is being stacked, and is equal to n where n is  
1448            the nth document in the job, starting with 1.  
1449  
1450            Implementations that only support one document jobs SHOULD  
1451            NOT implement this attribute.  
1452  
1453            jobCollationType(97),            JmJobCollationTypeTC  
1454            INTEGER: The type of job collation. See also Section 3.4,  
1455            entitled 'Monitoring Job Progress'.  
1456

```
1457 ++++++
1458 + Impression attributes
1459 +
1460 + See the definition of the terms 'impression', 'sheet',
1461 + and 'page' in Section 2.
1462 +
1463 + See also jmJobImpressionsPerCopyRequested and
1464 + jmJobImpressionsCompleted objects in the jmJobTable.
1465 ++++++
1466
1467 impressionsSpooled(110),          Integer32 (-2..2147483647)
1468     INTEGER: The number of impressions spooled to the server
1469     or device for the job so far.
1470
1471 impressionsSentToDevice(111),    Integer32 (-2..2147483647)
1472     INTEGER: The number of impressions sent to the device for
1473     the job so far.
1474
1475 impressionsInterpreted(112),     Integer32 (-2..2147483647)
1476     INTEGER: The number of impressions interpreted for the job
1477     so far.
1478
1479 impressionsCompletedCurrentCopy(113),
1480     Integer32 (-2..2147483647)
1481     INTEGER: The number of impressions completed by the device
1482     for the current copy of the current document so far. For
1483     printing, the impressions completed includes interpreting,
1484     marking, and stacking the output. For other types of job
1485     services, the number of impressions completed includes the
1486     number of impressions processed.
1487
1488     This value SHALL be reset to 0 for each document in the job
1489     and for each document copy.
1490
1491 fullColorImpressionsCompleted(114), Integer32 (-2..2147483647)
1492     INTEGER: The number of full color impressions completed by
1493     the device for this job so far. For printing, the
1494     impressions completed includes interpreting, marking, and
1495     stacking the output. For other types of job services, the
1496     number of impressions completed includes the number of
1497     impressions processed. Full color impressions are typically
1498     defined as those requiring 3 or more colorants, but this
1499     MAY vary by implementation. In any case, the value of this
1500     attribute counts by 1 for each side that has full color,
1501     not by the number of colors per side (and the other
1502     impression counters are incremented, except
1503     highlightColorImpressionsCompleted(115)).
1504
```

1505 highlightColorImpressionsCompleted(115),  
1506 Integer32 (-2..2147483647)  
1507 INTEGER: The number of highlight color impressions  
1508 completed by the device for this job so far. For printing,  
1509 the impressions completed includes interpreting, marking,  
1510 and stacking the output. For other types of job services,  
1511 the number of impressions completed includes the number of  
1512 impressions processed. Highlight color impressions are  
1513 typically defined as those requiring black plus one other  
1514 colorant, but this MAY vary by implementation. In any  
1515 case, the value of this attribute counts by 1 for each side  
1516 that has highlight color (and the other impression counters  
1517 are incremented, except  
1518 fullColorImpressionsCompleted(114)).  
1519  
1520 ++++++  
1521 + Page attributes  
1522 +  
1523 + See the definition of 'impression', 'sheet', and 'page'  
1524 + in Section 2.  
1525 ++++++

1526  
1527 pagesRequested(130), Integer32 (-2..2147483647)  
1528 INTEGER: The number of logical pages requested by the job  
1529 to be processed.  
1530

1531 pagesCompleted(131), Integer32 (-2..2147483647)  
1532 INTEGER: The number of logical pages completed for this  
1533 job so far.  
1534  
1535 For implementations where multiple copies are produced by  
1536 the interpreter with only a single pass over the data, the  
1537 final value SHALL be equal to the value of the  
1538 pagesRequested object. For implementations where multiple  
1539 copies are produced by the interpreter by processing the  
1540 data for each copy, the final value SHALL be a multiple of  
1541 the value of the pagesRequested object.  
1542

1543 NOTE - See the impressionsCompletedCurrentCopy and  
1544 pagesCompletedCurrentCopy attributes for attributes that  
1545 are reset on each document copy.  
1546

1547 NOTE - The pagesCompleted object can be used with the  
1548 pagesRequested object to provide an indication of the  
1549 relative progress of the job, provided that the  
1550 multiplicative factor is taken into account for some  
1551 implementations of multiple copies.  
1552

1553 pagesCompletedCurrentCopy(132), Integer32 (-2..2147483647)  
1554 INTEGER: The number of logical pages completed for the  
1555 current copy of the document so far. This value SHALL be  
1556 reset to 0 for each document in the job and for each  
1557 document copy.  
1558  
1559 ++++++  
1560 + Sheet attributes  
1561 +  
1562 + See the definition of 'impression', 'sheet', and 'page'  
1563 + in Section 2.  
1564 ++++++

1565  
1566 sheetsRequested(150), Integer32 (-2..2147483647)  
1567 INTEGER: The total number of medium sheets requested to be  
1568 produced for this job.  
1569  
1570 Unlike the jmJobKOctetsPerCopyRequested and  
1571 jmJobImpressionsPerCopyRequested attributes, the  
1572 sheetsRequested(150) attribute SHALL include the  
1573 multiplicative factor contributed by the number of copies  
1574 and so is the total number of sheets to be produced by the  
1575 job, as opposed to the size of the document(s) submitted.  
1576

1577 sheetsCompleted(151), Integer32 (-2..2147483647)  
1578 INTEGER: The total number of medium sheets that have  
1579 completed marking and stacking for the entire job so far  
1580 whether those sheets have been processed on one side or on  
1581 both.  
1582

1583 sheetsCompletedCurrentCopy(152), Integer32 (-2..2147483647)  
1584 INTEGER: The number of medium sheets that have completed  
1585 marking and stacking for the current copy of a document in  
1586 the job so far whether those sheets have been processed on  
1587 one side or on both.  
1588  
1589 The value of this attribute SHALL be 0 before the job  
1590 starts processing and SHALL be reset to 1 after the first  
1591 sheet of each document and document copy in the job is  
1592 processed and stacked.  
1593

```

1594 ++++++
1595 + Resources attributes (requested and consumed)
1596 +
1597 + Pairs of these attributes can be used by monitoring
1598 + applications to show an indication of relative usage to
1599 + users, i.e., a 'thermometer'.
1600 ++++++
1601
1602 mediumRequested(170),                               JmMediumTypeTC
1603                                                         AND/OR
1604                                                         JmJobStringTC (SIZE(0..63))
1605     INTEGER: MULTI-ROW: The type
1606     AND/OR
1607     OCTETS: MULTI-ROW: the name of the medium that is
1608     required by the job.
1609
1610     NOTE - The name (JmJobStringTC) values correspond to the
1611     name values of the prtInputMediaName object in the Printer
1612     MIB [print-mib] and the name, size, and input tray values
1613     of the IPP 'media' attribute [ipp-model].
1614
1615 mediumConsumed(171),                               Integer32 (-2..2147483647)
1616                                                         AND
1617                                                         JmJobStringTC (SIZE(0..63))
1618     INTEGER: MULTI-ROW: The number of sheets
1619     AND
1620     OCTETS: MULTI-ROW: the name of the medium that has been
1621     consumed so far whether those sheets have been processed on
1622     one side or on both.
1623
1624     This attribute SHALL have both Integer32 and OCTET STRING
1625     (represented as JmJobStringTC) values.
1626
1627     NOTE - The name (JmJobStringTC) values correspond to the
1628     name values of the prtInputMediaName object in the Printer
1629     MIB [print-mib] and the name, size, and input tray values
1630     of the IPP 'media' attribute [ipp-model].
1631
1632 colorantRequested(172),                             Integer32 (-2..2147483647)
1633                                                         AND/OR
1634                                                         JmJobStringTC (SIZE(0..63))
1635     INTEGER: MULTI-ROW: The index (prtMarkerColorantIndex) in
1636     the Printer MIB[print-mib]
1637     AND/OR
1638     OCTETS: MULTI-ROW: the name of the colorant requested.
1639
1640     NOTE - The name (JmJobStringTC) values correspond to the
1641     name values of the prtMarkerColorantValue object in the
1642     Printer MIB. Examples are: red, blue.

```



1643  
1644           colorantConsumed(173),                   Integer32 (-2..2147483647)  
1645    AND/OR  
1646    JmJobStringTC (SIZE(0..63))  
1647           INTEGER: MULTI-ROW: The index (prtMarkerColorantIndex) in  
1648           the Printer MIB[print-mib]  
1649           AND/OR  
1650           OCTETS: MULTI-ROW: the name of the colorant consumed.  
1651  
1652           NOTE - The name (JmJobStringTC) values correspond to the  
1653           name values of the prtMarkerColorantValue object in the  
1654           Printer MIB. Examples are: red, blue  
1655  
1656           mediumTypeConsumed(174),                   Integer32 (-2..2147483647)  
1657    AND  
1658    JmJobStringTC (SIZE(0..63))  
1659           INTEGER: MULTI-ROW: The number of sheets of the indicated  
1660           medium type that has been consumed so far whether those  
1661           sheets have been processed on one side or on both  
1662           AND  
1663           OCTETS: MULTI-ROW: the name of that medium type.  
1664  
1665           This attribute SHALL have both Integer32 and OCTET STRING  
1666           (represented as JmJobStringTC) values.  
1667  
1668           NOTE - The type name (JmJobStringTC) values correspond to  
1669           the type name values of the prtInputMediaType object in the  
1670           Printer MIB [print-mib]. Values are: 'stationery',  
1671           'transparency', 'envelope', etc. These medium type names  
1672           correspond to the enum values of JmMediumTypeTC used in the  
1673           mediumRequested attribute.  
1674  
1675           mediumSizeConsumed(175),                   Integer32 (-2..2147483647)  
1676    AND  
1677    JmJobStringTC (SIZE(0..63))  
1678           INTEGER: MULTI-ROW: The number of sheets of the indicated  
1679           medium size that has been consumed so far whether those  
1680           sheets have been processed on one side or on both  
1681           AND  
1682           OCTETS: MULTI-ROW: the name of that medium size.  
1683  
1684           This attribute SHALL have both Integer32 and OCTET STRING  
1685           (represented as JmJobStringTC) values.  
1686  
1687           NOTE - The size name (JmJobStringTC) values correspond to  
1688           the size name values in the Printer MIB [print-mib]  
1689           Appendix B. These size name values are also a subset of  
1690           the keyword values defined by [ipp-model] for the 'media'  
1691           Job Template attribute. Values are: 'letter', 'a', 'iso-  
1692           a4', 'jis-b4', etc.  
1693

```

1694 ++++++
1695 + Time attributes (set by server or device)
1696 +
1697 + This section of attributes are ones that are set by the
1698 + server or device that accepts jobs. Two forms of time are
1699 + provided. Each form is represented in a separate attribute.
1700 + See section 3.1.2 and section 3.1.3 for the
1701 + conformance requirements for time attribute for agents and
1702 + monitoring applications, respectively. The two forms are:
1703 +
1704 + 'DateAndTime' is an 8 or 11 octet binary encoded year,
1705 + month, day, hour, minute, second, deci-second with
1706 + optional offset from UTC. See SNMPv2-TC [SMIV2-TC].
1707 +
1708 + NOTE: 'DateAndTime' is not printable characters; it is
1709 + binary.
1710 +
1711 + 'JmTimeStampTC' is the time of day measured in the number of
1712 + seconds since the system was booted.
1713 ++++++
1714
1715 jobSubmissionToServerTime(190),      JmTimeStampTC
1716                                     AND/OR
1717                                     DateAndTime
1718     INTEGER: Configuration 3 only: The time
1719     AND/OR
1720     OCTETS: the date and time that the job was submitted to
1721     the server (as distinguished from the device which uses
1722     jobSubmissionTime).
1723
1724 jobSubmissionTime(191),              JmTimeStampTC
1725                                     AND/OR
1726                                     DateAndTime
1727     INTEGER: Configurations 1, 2, and 3: The time
1728     AND/OR
1729     OCTETS: the date and time that the job was submitted to
1730     the server or device to which the agent is providing
1731     access.
1732
1733 jobStartedBeingHeldTime(192),       JmTimeStampTC
1734                                     AND/OR
1735                                     DateAndTime
1736     INTEGER: The time
1737     AND/OR
1738     OCTETS: the date and time that the job last entered the
1739     pendingHeld state. If the job has never entered the
1740     pendingHeld state, then the value SHALL be '0' or the
1741     attribute SHALL not be present in the table.

```

1742  
1743           jobStartedProcessingTime(193),        JmTimeStampTC  
1744    AND/OR  
1745    DateAndTime  
1746            INTEGER:   The time  
1747            AND/OR  
1748            OCTETS:   the date and time that the job started processing.  
1749  
1750           jobCompletionTime(194),                JmTimeStampTC  
1751    AND/OR  
1752    DateAndTime  
1753            INTEGER:   The time  
1754            AND/OR  
1755            OCTETS:   the date and time that the job entered the  
1756                      completed, canceled, or aborted state.  
1757  
1758           jobProcessingCPUtime(195)               Integer32 (-2..2147483647)  
1759    UNITS        'seconds'  
1760            INTEGER:   The amount of CPU time in seconds that the job  
1761                      has been in the processing state.  If the job enters the  
1762                      processingStopped state, that elapsed time SHALL not be  
1763                      included.  In other words, the jobProcessingCPUtime value  
1764                      SHOULD be relatively repeatable when the same job is  
1765                      processed again on the same device.

1766

1767 **3.4 Monitoring Job Progress**

1768 There are a number of objects and attributes for monitoring the  
1769 progress of a job. These objects and attributes count the number of K  
1770 octets, impressions, sheets, and pages requested or completed. For  
1771 impressions and sheets, "completed" means stacked, unless the  
1772 implementation is unable to detect when each sheet is stacked, in which  
1773 case stacked is approximated when processing of each sheet completes.  
1774 There are objects and attributes for the overall job and for the  
1775 current copy of the document currently being stacked. For the latter,  
1776 the rate at which the various objects and attributes count depends on  
1777 the sheet and document collation of the job.

1778 Job Collation included sheet collation and document collation. Sheet  
1779 collation is defined to be the ordering of sheets within a document  
1780 copy. Document collation is defined to be ordering of document copies  
1781 within a multi-document job. There are three types of job collation  
1782 (see terminology definitions in Section 2):

1783 1. uncollatedSheets(3) - No collation of the sheets within each  
1784 document copy, i.e., each sheet of a document that is to  
1785 produce multiple copies is replicated before the next sheet in  
1786 the document is processed and stacked. If the device has an  
1787 output bin collator, the uncollatedSheets(3) value may actually  
1788 produce collated sheets as far as the user is concerned (in the  
1789 output bins). However, when the job collation is the  
1790 'uncollatedSheets(3)' value, job progress is indistinguishable  
1791 to a monitoring application between a device that has an output  
1792 bin collator and one that does not.

1793 2. collatedDocuments(4) - Collation of the sheets within each  
1794 document copy is performed within the printing device by making  
1795 multiple passes over either the source or an intermediate  
1796 representation of the document. In addition, when there are  
1797 multiple documents per job, the i'th copy of each document is  
1798 stacked before the j'th copy of each document, i.e., the  
1799 documents are collated within each job copy. For example, if a  
1800 job is submitted with documents, A and B, the job is made  
1801 available to the end user as: A, B, A, B, .... The  
1802 'collatedDocuments(4)' value corresponds to the IPP [ipp-model]  
1803 'separate-documents-collated-copies' value of the "multiple-  
1804 document-handling" attribute.  
1805

1806 If jobCopiesRequested or documentCopiesRequested = 1, then  
1807 jobCollationType is defined as 4.

1808 3. uncollatedDocuments(5) - Collation of the sheets within each  
1809 document copy is performed within the printing device by making  
1810 multiple passes over either the source or an intermediate  
1811 representation of the document. In addition, when there are

1812 multiple documents per job, all copies of the first document in  
1813 the job are stacked before the any copied of the next document  
1814 in the job, i.e., the documents are uncollated within the job.  
1815 For example, if a job is submitted with documents, A and B, the  
1816 job is mad available to the end user as: A, A, ..., B, B, ....  
1817 The 'uncollatedDocuments(5)' value corresponds to the IPP [ipp-  
1818 model] 'separate-documents-uncollated-copies' value of the  
1819 "multiple-document-handling" attribute.

1820 Consider the following four variables that are used to monitor the  
1821 progress of a job's impressions:

- 1822 1. jmJobImpressionsCompleted - counts the total number of  
1823 impressions stacked for the job
- 1824 2. impressionsCompletedCurrentCopy - counts the number of  
1825 impressions stacked for the current document copy
- 1826 3. sheetCompletedCopyNumber - identifies the number of the copy  
1827 for the current document being stacked where the first copy is  
1828 1.
- 1829 4. sheetCompletedDocumentNumber - identifies the current document  
1830 within the job that is being stacked where the first document  
1831 in a job is 1. NOTE: this attribute SHOULD NOT be implemented  
1832 for implementations that only support one document per job.

1833 For each of the three types of job collation, a job with three copies  
1834 of two documents (1, 2), where each document consists of 3 impressions,  
1835 the four variables have the following values as each sheet is stacked  
1836 for one-sided printing:

1837 Job Collation Type = uncollatedSheets(3)

1838

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

1839

1840 Job Collation Type = collatedDocuments(4)

1841

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

1842

1843 Job Collation Type = uncollatedDocuments(5)  
 1844

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

1845

### 1846 3.5 Job Identification

1847 There are a number of attributes that permit a user, operator or system  
 1848 administrator to identify jobs of interest, such as jobURI, jobName,  
 1849 jobOriginatingHost, etc. In addition, there is a jmJobSubmissionID  
 1850 object that is a text string table index. Being a table index allows a  
 1851 monitoring application to quickly locate and identify a particular job  
 1852 of interest that was submitted from a particular client by the user  
 1853 invoking the monitoring application without having to scan the entire  
 1854 job table. The Job Monitoring MIB needs to provide for identification  
 1855 of the job at both sides of the job submission process. The primary  
 1856 identification point is the client side. The jmJobSubmissionID allows  
 1857 the monitoring application to identify the job of interest from all the  
 1858 jobs currently "known" by the server or device. The value of  
 1859 jmJobSubmissionID can be assigned by either the client's local system  
 1860 or a downstream server or device. The point of assignment depends on  
 1861 the job submission protocol in use.

1862 The server/device-side identifier, called the jmJobIndex object, SHALL  
 1863 be assigned by the SNMP Job Monitoring MIB agent when the server or  
 1864 device accepts the jobs from submitting clients. The jmJobIndex object  
 1865 allows the interested party to obtain all objects desired that relate  
 1866 to a particular job. See Section 3.2, entitled 'The Job Tables and the



1867 Oldest Active and Newest Active Indexes' for the specification of how  
1868 the agent SHALL assign the jmJobIndex values.

1869 The MIB provides a mapping table that maps each jmJobSubmissionID value  
1870 to a corresponding jmJobIndex value generated by the agent, so that an  
1871 application can determine the correct value for the jmJobIndex value  
1872 for the job of interest in a single Get operation, given the Job  
1873 Submission ID. See the jmJobIDGroup.

1874 In some configurations there may be more than one application program  
1875 that monitors the same job when the job passes from one network entity  
1876 to another when it is submitted. See configuration 3. When there are  
1877 multiple job submission IDs, each entity MAY supply an appropriate  
1878 jmJobSubmissionID value. In this case there would be a separate entry  
1879 in the jmJobSubmissionID table, one for each jmJobSubmissionID. All  
1880 entries would map to the same jmJobIndex that contains the job data.  
1881 When the job is deleted, it is up to the agent to remove all entries  
1882 that point to the job from the jmJobSubmissionID table as well.

1883 The jobName attribute provides a name that the user supplies as a job  
1884 attribute with the job. The jobName attribute is not necessarily  
1885 unique, even for one user, let alone across users.

### 1886 3.6 Internationalization Considerations

1887 This section describes the internationalization considerations included  
1888 in this MIB.

#### 1889 3.6.1 Text generated by the server or device

1890 There are a few objects and attributes generated by the server or  
1891 device that SHALL be represented using the Universal Multiple-Octet  
1892 Coded Character Set (UCS) [ISO-10646]. These objects and attributes  
1893 are always supplied (if implemented) by the agent, not by the job  
1894 submitting client:

- 1895 1. jmGeneralJobSetName object
- 1896 2. processingMessage(6) attribute
- 1897 3. physicalDevice(32) (name value) attribute

1898 The character encoding scheme for representing these objects and  
1899 attributes SHALL be UTF-8 as recommended by RFC 2130 [RFC 2130] and the  
1900 "IETF Policy on Character Sets and Language" [char-set policy]. The  
1901 'JmUTF8StringTC' textual convention is used to indicate UTF-8 text  
1902 strings.

1903 NOTE - For strings in 7-bit US-ASCII, there is no impact since the UTF-  
1904 8 representation of 7-bit ASCII is identical to the US-ASCII [US-ASCII]  
1905 encoding.

1906 The text contained in the processingMessage(6) attribute is generated  
1907 by the server/device. The natural language for the

1908 processingMessage(6) attribute is identified by the  
1909 processingMessageNaturalLangTag(7) attribute. The  
1910 processingMessageNaturalLangTag(7) attribute uses the  
1911 JmNaturalLanguageTagTC textual convention which SHALL conform to the  
1912 language tag mechanism specified in RFC 1766 [RFC-1766]. The  
1913 JmNaturalLanguageTagTC value is the same as the IPP [IPP-model]  
1914 'naturalLanguage' attribute syntax. RFC 1766 specifies that a US-ASCII  
1915 string consisting of the natural language followed by an optional  
1916 country field. Both fields use the same two-character codes from ISO  
1917 639 [ISO-639] and ISO 3166 [ISO-3166], respectively, that are used in  
1918 the Printer MIB for identifying language and country.

1919 Examples of the values of the processingMessageNaturalLangTag(7)  
1920 attribute include:

- 1921 1. 'en' for English
- 1922 2. 'en-us' for US English
- 1923 3. 'fr' for French
- 1924 4. 'de' for German

### 1925 3.6.2 Text supplied by the job submitter

1926 All of the objects and attributes represented by the 'JmJobStringTC'  
1927 textual-convention are either (1) supplied in the job submission  
1928 protocol by the client that submits the job to the server or device or  
1929 (2) are defaulted by the server or device if the job submitting client  
1930 does not supply values. The agent SHALL represent these objects and  
1931 attributes in the MIB either (1) in the coded character set as they  
1932 were submitted or (2) MAY convert the coded character set to another  
1933 coded character set or encoding scheme. In any case, the resulting  
1934 coded character set representation SHOULD be UTF-8 [UTF-8], but SHALL  
1935 be one in which the code positions from 0 to 31 is not used, 32 to 127  
1936 is US-ASCII [US-ASCII], 127 is not unused, and the remaining code  
1937 positions 128 to 255 represent single-byte or multi-byte graphic  
1938 characters structured according to ISO 2022 [ISO 2022] or are unused.

1939 The coded character set SHALL be one of the ones registered with IANA  
1940 [IANA] and SHALL be identified by the jobCodedCharSet attribute in the  
1941 jmJobAttributeTable for the job. If the agent does not know what coded  
1942 character set was used by the job submitting client, the agent SHALL  
1943 either (1) return the 'unknown(2)' value for the jobCodedCharSet  
1944 attribute or (2) not return the jobCodedCharSet attribute for the job.

1945 Examples of coded character sets which meet this criteria for use as  
1946 the value of the jobCodedCharSet job attribute are: US-ASCII [US-  
1947 ASCII], ISO 8859-1 (Latin-1) [ISO 8859-1], any ISO 8859-n, HP Roman8,  
1948 IBM Code Page 850, Windows Default 8-bit set, UTF-8 [UTF-8], US-ASCII  
1949 plus JIS X0208-1990 Japanese [JIS X0208], US-ASCII plus GB2312-1980 PRC  
1950 Chinese [GB2312]. See the IANA registry of coded character sets [IANA  
1951 charsets].

1952 Examples of coded character sets which do not meet this criteria are:  
1953 national 7-bit sets conforming to ISO 646 (except US-ASCII), EBCDIC,  
1954 and ISO 10646 (Unicode) [ISO-10646]. In order to represent Unicode  
1955 characters, the UTF-8 [UTF-8] encoding scheme SHALL be used which has  
1956 been assigned the MIBenum value of '106' by IANA.

1957 The jobCodedCharSet attribute uses the imported 'CodedCharSet' textual-  
1958 convention from the Printer MIB [printmib].

1959 The natural language for attributes represented by the textual-  
1960 convention JmJobStringTC is identified either (1) by the  
1961 jobNaturalLanguageTag(9) attribute or is keywords in US-English (as in  
1962 IPP). A monitoring application SHOULD attempt to localize keywords  
1963 into the language of the user by means of some lookup mechanism. If  
1964 the keyword value is not known to the monitoring application, the  
1965 monitoring application SHOULD assume that the value is in the natural  
1966 language specified by the job's jobNaturalLanguageTag(9) attribute and  
1967 SHOULD present the value to its user as is. The  
1968 jobNaturalLanguageTag(9) attribute value SHALL have the same syntax and  
1969 semantics as the processingMessageNaturalLangTag(7) attribute, except  
1970 that the jobNaturalLanguageTag(9) attribute identifies the natural  
1971 language of attributes supplied by the job submitter instead of the  
1972 natural language of the processingMessage(6) attribute. See Section  
1973 3.6.1.

1974 3.6.3 'DateAndTime' for representing the date and time

1975 This MIB also contains objects that are represented using the  
1976 DateAndTime textual convention from SMIV2 [SMIV2-TC]. The job  
1977 management application SHALL display such objects in the locale of the  
1978 user running the monitoring application.

### 1979 **3.7 IANA and PWG Registration Considerations**

1980 This MIB does not require any additional registration schemes for IANA,  
1981 but does depend on registration schemes that other Internet standards  
1982 track specifications have set up. The names of these IANA registration  
1983 assignments under the /in-notes/iana/assignments/ path:

1984 1. printer-language-numbers - used as enums in the documentFormat(38)  
1985 attribute

1986 2. media-types - uses as keywords in the documentFormat(38) attribute

1987 3. character-sets - used as enums in the jobCodedCharSet(8) attribute

1988 The Printer Working Group (PWG) will handle registration of additional  
1989 enums after approving this standard, according to the procedures  
1990 described in this section:

1991

## 1992 3.7.1 PWG Registration of enums

1993 This specification uses textual conventions to define enumerated values  
1994 (enums) and bit values. Enumerations (enums) and bit values are sets  
1995 of symbolic values defined for use with one or more objects or  
1996 attributes. All enumeration sets and bit value sets are assigned a  
1997 symbolic data type name (textual convention). As a convention the  
1998 symbolic name ends in "TC" for textual convention. These enumerations  
1999 are defined at the beginning of the MIB module specification.

2000 The PWG has defined several type of enumerations for use in the Job  
2001 Monitoring MIB and the Printer MIB[print-mib]. These types differ in  
2002 the method employed to control the addition of new enumerations.  
2003 Throughout this document, references to "type n enum", where n can be  
2004 1, 2 or 3 can be found in the various tables. The definitions of these  
2005 types of enumerations are:

## 2006 3.7.1.1 Type 1 enumerations

2007 Type 1 enumeration: All the values are defined in the Job Monitoring  
2008 MIB specification (RFC for the Job Monitoring MIB). Additional  
2009 enumerated values require a new RFC.

2010 There are no type 1 enums in the current draft.

## 2011 3.7.1.2 Type 2 enumerations

2012 Type 2 enumeration: An initial set of values are defined in the Job  
2013 Monitoring MIB specification. Additional enumerated values are  
2014 registered with the PWG.

2015 The following type 2 enums are contained in the current draft :

- 2016 1. JmUTF8StringTC
- 2017 2. JmJobStringTC
- 2018 3. JmNaturalLanguageTagTC
- 2019 4. JmTimeStampTC
- 2020 5. JmFinishingTC [same enum values as IPP "finishing" attribute]
- 2021 6. JmPrintQualityTC [same enum values as IPP "print-quality"
- 2022 attribute]
- 2023 7. JmTonerEconomyTC
- 2024 8. JmMediumTypeTC
- 2025 9. JmJobSubmissionIDTypeTC
- 2026 10. JmJobCollationTypeTC
- 2027 11. JmJobStateTC [same enum values as IPP "job-state" attribute]
- 2028 12. JmAttributeTypeTC

2029 For those textual conventions that have the same enum values as the  
2030 indicated IPP Job attribute are simultaneously registered by the PWG  
2031 for use with IPP [ipp-model] and the Job Monitoring MIB.

## 2032 3.7.1.3 Type 3 enumeration

2033 Type 3 enumeration: An initial set of values are defined in the Job  
2034 Monitoring MIB specification. Additional enumerated values are  
2035 registered through the PWG without PWG review.

2036 There are no type 3 enums in the current draft.

## 2037 3.7.2 PWG Registration of type 2 bit values

2038 This draft contains the following type 2 bit value textual-conventions:

- 2039 1. JmJobServiceTypesTC
- 2040 2. JmJobStateReasons1TC
- 2041 3. JmJobStateReasons2TC
- 2042 4. JmJobStateReasons3TC
- 2043 5. JmJobStateReasons4TC

2044 These textual-conventions are defined as bits in an Integer so that  
2045 they can be used with SNMPv1 SMI. The jobStateReasonsN (N=1..4)  
2046 attributes are defined as bit values using the corresponding  
2047 JmJobStateReasonsMTC textual-conventions.

2048 The registration of JmJobServiceTypesTC and JmJobStateReasonsMTC bit  
2049 values follow the procedures for a type 2 enum as specified in Section  
2050 3.7.1.2.

## 2051 3.7.3 PWG Registration of Job Submission Id Formats

2052 In addition to enums and bit values, this specification assigns a  
2053 single ASCII digit or letter to various job submission ID formats. See  
2054 the JmJobSubmissionIDTypeTC textual-convention and the object. The  
2055 registration of JobSubmissionID format numbers follows the procedures  
2056 for a type 2 enum as specified in Section 3.7.1.2.

## 2057 3.7.4 PWG Registration of MIME types/sub-types for document-formats

2058 The documentFormat(38) attribute has MIME type/sub-type values for  
2059 indicating document formats which IANA registers as "media type" names.  
2060 The values of the documentFormat(38) attribute are the same as the  
2061 corresponding Internet Printing Protocol (IPP) "document-format" Job  
2062 attribute values [ipp-model].

2063 **3.8 Security Considerations**

2064 3.8.1 Read-Write objects

2065 All objects are read-only, greatly simplifying the security  
2066 considerations. If another MIB augments this MIB, that MIB might  
2067 accept SNMP Write operations to objects in that MIB whose effect is to  
2068 modify the values of read-only objects in this MIB. However, that MIB  
2069 SHALL have to support the required access control in order to achieve  
2070 security, not this MIB.

2071 3.8.2 Read-Only Objects In Other User's Jobs

2072 The security policy of some sites MAY be that unprivileged users can  
2073 only get the objects from jobs that they submitted, plus a few minimal  
2074 objects from other jobs, such as the jmJobKOctetsPerCopyRequested and  
2075 jmJobKOctetsProcessed objects, so that a user can tell how busy a  
2076 printer is. Other sites MAY allow all unprivileged users to see all  
2077 objects of all jobs. This MIB does not require, nor does it specify  
2078 how, such restrictions would be implemented. A monitoring application  
2079 SHOULD enforce the site security policy with respect to returning  
2080 information to an unprivileged end user that is using the monitoring  
2081 application to monitor jobs that do not belong to that user, i.e., the  
2082 jmJobOwner object in the jmJobTable does not match the user's user  
2083 name.

2084 An operator is a privileged user that would be able to see all objects  
2085 of all jobs, independent of the policy for unprivileged users.

2086 **3.9 Notifications**

2087 This MIB does not specify any notifications. For simplicity,  
2088 management applications are expected to poll for status. The  
2089 jmGeneralJobPersistence and jmGeneralAttributePersistence objects  
2090 assist an application to determine the polling rate. The resulting  
2091 network traffic is not expected to be significant.

2092 4 MIB specification

2093 The following pages constitute the actual Job Monitoring MIB.

```
2094 Job-Monitoring-MIB DEFINITIONS ::= BEGIN
2095
2096 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

2097
2098 -- Use the enterprises arc assigned to the PWG which is pwg(2699).
2099 -- Group all PWG mibs under mibs(1).
2100
2101 jobmonMIB MODULE-IDENTITY
2102     LAST-UPDATED "9810020000Z"
2103     ORGANIZATION "Printer Working Group (PWG)"
2104     CONTACT-INFO
2105         "Tom Hastings
2106         Postal:  Xerox Corp.
2107                 Mail stop ESAE-231
2108                 701 S. Aviation Blvd.
2109                 El Segundo, CA 90245
2110
2111         Tel:      (301)333-6413
2112         Fax:      (301)333-5514
2113         E-mail:   hastings@cpl0.es.xerox.com
2114
2115         Send questions and comments to the Printer Working Group (PWG)
2116         using the Job Monitoring Project (JMP) Mailing List:
2117         jmp@pwg.org
2118
2119         For further information, including how to subscribe to the
2120         jmp mailing list, access the PWG web page under 'JMP':
2121
2122         http://www.pwg.org/
2123
2124         Implementers of this specification are encouraged to join the
2125         jmp mailing list in order to participate in discussions on any
2126         clarifications needed and registration proposals being reviewed
2127         in order to achieve consensus."
2128     DESCRIPTION
2129         "The MIB module for monitoring job in servers, printers, and
2130         other devices.
2131
2132         Version: 1.2"
2133     ::= { enterprises pwg(2699) mibs(1) jobmonMIB(1) }
```

```
2134
2135 -- Textual conventions for this MIB module
2136
2137 JmUTF8StringTC ::= TEXTUAL-CONVENTION
2138     DISPLAY-HINT "255a"
2139     STATUS      current
2140     DESCRIPTION
2141         "To facilitate internationalization, this TC represents
2142         information taken from the ISO/IEC IS 10646-1 character set,
2143         encoded as an octet string using the UTF-8 character encoding
2144         scheme.
2145
2146         See section 3.6.1, entitled: 'Text generated by the server or
2147         device'."
2148     SYNTAX      OCTET STRING (SIZE (0..63))
2149
2150
2151
2152
2153 JmJobStringTC ::= TEXTUAL-CONVENTION
2154     STATUS      current
2155     DESCRIPTION
2156         "To facilitate internationalization, this TC represents
2157         information using any coded character set registered by IANA as
2158         specified in section 3.7. While it is recommended that the
2159         coded character set be UTF-8 [UTF-8], the actual coded
2160         character set SHALL be indicated by the value of the
2161         jobCodedCharSet(8) attribute for the job.
2162
2163         See section 3.6.2, entitled: 'Text supplied by the job
2164         submitter'."
2165     SYNTAX      OCTET STRING (SIZE (0..63))
2166
2167
2168
2169
2170 JmNaturalLanguageTagTC ::= TEXTUAL-CONVENTION
2171     STATUS      current
2172     DESCRIPTION
2173         "An IETF RFC 1766-compliant 'language tag', with zero or more
2174         sub-tags that identify a natural language. While RFC 1766
2175         specifies that the US-ASCII values are case-insensitive, this
2176         MIB specification requires that all characters SHALL be lower
2177         case in order to simplify comparing by management applications.
2178
2179         See section 3.6.1, entitled: 'Text generated by the server or
2180         device' and section 3.6.2, entitled: 'Text supplied by the job
2181         submitter'."
2182     SYNTAX      OCTET STRING (SIZE (0..63))
2183
2184
2185
2185 JmTimeStampTC ::= TEXTUAL-CONVENTION
```



```
2186     STATUS      current
2187     DESCRIPTION
2188         "The simple time at which an event took place.  The units are
2189         in seconds since the system was booted.
2190
2191         NOTE - JmTimeStampTC is defined in units of seconds, rather
2192         than 100ths of seconds, so as to be simpler for agents to
2193         implement (even if they have to implement the 100ths of a
2194         second to comply with implementing sysUpTime in MIB-II[mib-
2195         II].)
2196
2197         NOTE - JmTimeStampTC is defined as an Integer32 so that it can
2198         be used as a value of an attribute, i.e., as a value of the
2199         jmAttributeValueAsInteger object.  The TimeStamp textual-
2200         convention defined in SNMPv2-TC [SMIV2-TC] is defined as an
2201         APPLICATION 3 IMPLICIT INTEGER tag, not an Integer32 which is
2202         defined in SNMPv2-SMI [SMIV2-TC] as UNIVERSAL 2 IMPLICIT
2203         INTEGER, so cannot be used in this MIB as one of the values of
2204         jmAttributeValueAsInteger."
2205     SYNTAX      INTEGER (0..2147483647)
2206
2207
2208
2209
2210 JmJobSourcePlatformTypeTC ::= TEXTUAL-CONVENTION
2211     STATUS      current
2212     DESCRIPTION
2213         "The source platform type that can submit jobs to servers or
2214         devices in any of the 3 configurations.
2215
2216         This is a type 2 enumeration.  See Section 3.7.1.2.  See also
2217         IANA operating-system-names registry."
2218     SYNTAX      INTEGER {
2219         other(1),
2220         unknown(2),
2221         sptUNIX(3),           -- UNIX
2222         sptOS2(4),           -- OS/2
2223         sptPCDOS(5),         -- DOS
2224         sptNT(6),           -- NT
2225         sptMVS(7),          -- MVS
2226         sptVM(8),           -- VM
2227         sptOS400(9),         -- OS/400
2228         sptVMS(10),         -- VMS
2229         sptWindows(11),     -- Windows
2230         sptNetWare(12)      -- NetWare
2231     }
```

```
2221
2222 JmFinishingTC ::= TEXTUAL-CONVENTION
2223     STATUS      current
2224     DESCRIPTION
2225         "The type of finishing operation.
2226
2227         These values are the same as the enum values of the IPP
2228         'finishings' attribute.  See Section 3.7.1.2.
2229
2230         other(1),
2231             Some other finishing operation besides one of the specified
2232             or registered values.
2233
2234         unknown(2),
2235             The finishing is unknown.
2236
2237         none(3),
2238             Perform no finishing.
2239
2240         staple(4),
2241             Bind the document(s) with one or more staples. The exact
2242             number and placement of the staples is site-defined.
2243
2244         punch(5),
2245             This value indicates that holes are required in the
2246             finished document. The exact number and placement of the
2247             holes is site-defined. The punch specification MAY be
2248             satisfied (in a site- and implementation-specific manner)
2249             either by drilling/punching, or by substituting pre-drilled
2250             media.
2251
2252         cover(6),
2253             This value is specified when it is desired to select a non-
2254             printed (or pre-printed) cover for the document. This does
2255             not supplant the specification of a printed cover (on cover
2256             stock medium) by the document itself.
2257
2258         bind(7)
2259             This value indicates that a binding is to be applied to the
2260             document; the type and placement of the binding is product-
2261             specific.
2262
2263         This is a type 2 enumeration.  See Section 3.7.1.2."
2264     SYNTAX      INTEGER {
2265         other(1),
2266         unknown(2),
2267         none(3),
2268         staple(4),
2269         punch(5),
2270         cover(6),
2271         bind(7)
2272     }
```

```
2273
2274
2275 JmPrintQualityTC ::= TEXTUAL-CONVENTION
2276     STATUS      current
2277     DESCRIPTION
2278         "Print quality settings.
2279
2280         These values are the same as the enum values of the IPP 'print-
2281         quality' attribute.  See Section 3.7.1.2.
2282
2283         This is a type 2 enumeration.  See Section 3.7.1.2."
2284     SYNTAX      INTEGER {
2285         other(1),      -- Not one of the specified or registered
2286                       -- values.
2287         unknown(2),   -- The actual value is unknown.
2288         draft(3),     -- Lowest quality available on the printer.
2289         normal(4),    -- Normal or intermediate quality on the
2290                       -- printer.
2291         high(5)       -- Highest quality available on the printer.
2292     }
2293
2294
2295 JmPrinterResolutionTC ::= TEXTUAL-CONVENTION
2296     STATUS      current
2297     DESCRIPTION
2298         "Printer resolutions.
2299
2300         Nine octets consisting of two 4-octet SIGNED-INTEGERS followed
2301         by a SIGNED-BYTE.  The values are the same as those specified
2302         in the Printer MIB [printmib].  The first SIGNED-INTEGER
2303         contains the value of prtMarkerAddressabilityXFeedDir.  The
2304         second SIGNED-INTEGER contains the value of
2305         prtMarkerAddressabilityFeedDir.  The SIGNED-BYTE contains the
2306         value of prtMarkerAddressabilityUnit.
2307
2308         Note: the latter value is either 3 (tenThousandsOfInches) or 4
2309         (micrometers) and the addressability is in 10,000 units of
2310         measure.  Thus the SIGNED-INTEGERS represent integral values in
2311         either dots-per-inch or dots-per-centimeter.
2312
2313         The syntax is the same as the IPP 'printer-resolution'
2314         attribute.  See Section 3.7.1.2."
2315     SYNTAX      OCTET STRING (SIZE(9))
2316
```

```
2312
2313 JmTonerEconomyTC ::= TEXTUAL-CONVENTION
2314     STATUS      current
2315     DESCRIPTION
2316         "Toner economy settings.
2317
2318         This is a type 2 enumeration.  See Section 3.7.1.2."
2319     SYNTAX      INTEGER {
                unknown(2),      -- unknown.
                off(3),          -- Off. Normal. Use full toner.
                on(4)            -- On. Use less toner than normal.
                }
2320
2321
2322
2323
2324 JmBooleanTC ::= TEXTUAL-CONVENTION
2325     STATUS      current
2326     DESCRIPTION
2327         "Boolean true or false value.
2328
2329         This is a type 2 enumeration.  See Section 3.7.1.2."
2330     SYNTAX      INTEGER {
                unknown(2),      -- unknown.
                false(3),        -- FALSE.
                true(4)          -- TRUE.
                }
2331
2332
2333
2334
2335 JmMediumTypeTC ::= TEXTUAL-CONVENTION
2336     STATUS      current
2337     DESCRIPTION
2338         "Identifies the type of medium.
2339
2340         other(1),
2341             The type is neither one of the values listed in this
2342             specification nor a registered value.
2343
2344         unknown(2),
2345             The type is not known.
2346
2347         stationery(3),
2348             Separately cut sheets of an opaque material.
2349
2350         transparency(4),
2351             Separately cut sheets of a transparent material.
2352
2353         envelope(5),
2354             Envelopes that can be used for conventional mailing
2355             purposes.
```

2356  
2357       envelopePlain(6),  
2358            Envelopes that are not preprinted and have no windows.  
2359  
2360       envelopeWindow(7),  
2361            Envelopes that have windows for addressing purposes.  
2362  
2363       continuousLong(8),  
2364            Continuously connected sheets of an opaque material  
2365            connected along the long edge.  
2366  
2367       continuousShort(9),  
2368            Continuously connected sheets of an opaque material  
2369            connected along the short edge.  
2370  
2371       tabStock(10),  
2372            Media with tabs.  
2373  
2374       multiPartForm(11),  
2375            Form medium composed of multiple layers not pre-attached to  
2376            one another; each sheet MAY be drawn separately from an  
2377            input source.  
2378  
2379       labels(12),  
2380            Label-stock.  
2381  
2382       multiLayer(13)  
2383            Form medium composed of multiple layers which are pre-  
2384            attached to one another, e.g. for use with impact printers.  
2385  
2386       This is a type 2 enumeration. See Section 3.7.1.2. These enum  
2387       values correspond to the keyword name strings of the  
2388       prtInputMediaType object in the Printer MIB [print-mib]. There  
2389       is no printer description attribute in IPP/1.0 that represents  
2390       these values."  
2391       SYNTAX        INTEGER {  
2392            other(1),  
2393            unknown(2),  
2394            stationery(3),  
2395            transparency(4),  
2396            envelope(5),  
2397            envelopePlain(6),  
2398            envelopeWindow(7),  
2399            continuousLong(8),  
2400            continuousShort(9),  
2401            tabStock(10),  
2402            multiPartForm(11),  
2403            labels(12),  
2404            multiLayer(13)  
2405        }  
2406  
2407

```
2408 JmJobCollationTypeTC ::= TEXTUAL-CONVENTION
2409     STATUS      current
2410     DESCRIPTION
2411         "This value is the type of job collation. Implementations that
2412         don't support multiple documents or don't support multiple
2413         copies SHALL NOT support the uncollatedDocuments(5) value.
2414
2415         This is a type 2 enumeration. See Section 3.7.1.2. See also
2416         Section 3.4, entitled 'Monitoring Job Progress'."
2417     SYNTAX      INTEGER {
2418         other(1),
2419         unknown(2),
2420         uncollatedSheets(3),      -- sheets within each document copy
2421                                   -- are not collated: 1 1 ..., 2 2 ...,
2422                                   -- No corresponding value of IPP
2423                                   -- "multiple-document-handling"
2424         collatedDocuments(4),    -- internal collated sheets,
2425                                   -- documents: A, B, A, B, ...
2426                                   -- Corresponds to IPP "multiple-
2427                                   -- document-handling"='separate-
2428                                   -- documents-collated-copies'
2429         uncollatedDocuments(5)  -- internal collated sheets,
2430                                   -- documents: A, A, ..., B, B, ...
2431                                   -- Corresponds to IPP "multiple-
2432                                   -- document-handling"='separate-
2433                                   -- documents-uncollated-copies'
2434     }
2435
2436
2437 JmJobSubmissionIDTypeTC ::= TEXTUAL-CONVENTION
2438     STATUS      current
2439     DESCRIPTION
2440         "Identifies the format type of a job submission ID.
2441
2442         Each job submission ID is a fixed-length, 48-octet printable
2443         US-ASCII [US-ASCII] coded character string containing no
2444         control characters, consisting of the following fields:
2445
2446         octet 1: The format letter identifying the format. The US-
2447         ASCII characters '0-9', 'A-Z', and 'a-z' are assigned in
2448         order giving 62 possible formats.
2449         octets 2-40: A 39-character, US-ASCII trailing SPACE filled
2450         field specified by the format letter, if the data is less
2451         than 39 ASCII characters.
2452         octets 41-48: A sequential or random US-ASCII number to make
2453         the ID quasi-unique.
2454
2455         If the client does not supply a job submission ID in the job
2456         submission protocol, then the agent SHALL assign a job
2457         submission ID using any of the standard formats that are
2458         reserved for the agent. Clients SHALL not use formats that are
2459         reserved for agents and agents SHALL NOT use formats that are
```

2460 reserved for clients, in order to reduce conflicts in ID  
 2461 generation. See the description for which formats are reserved  
 2462 for clients or for agents.  
 2463

2464 Registration of additional formats may be done following the  
 2465 procedures described in Section 3.7.3.  
 2466

2467 The format values defined at the time of completion of this  
 2468 specification are:  
 2469

2470 Format

2471 Letter Description

2472 -----

2473 '0' Job Owner generated by the server/device  
 2474 octets 2-40: The last 39 bytes of the jmJobOwner object.  
 2475 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
 2476 assigned by the agent.  
 2477 This format is reserved for agents.  
 2478

2479 NOTE - Clients wishing to use a job submission ID that  
 2480 incorporates the job owner, SHALL use format '8', not  
 2481 format '0'.  
 2482

2483 '1' Job Name

2484 octets 2-40: The last 39 bytes of the jobName attribute.  
 2485 octets 41-48: The US-ASCII 8-decimal-digit random number  
 2486 assigned by the client.  
 2487 This format is reserved for clients.  
 2488

2489 '2' Client MAC address

2490 octets 2-40: The client MAC address: in hexadecimal with each  
 2491 nibble of the 6 octet address being '0'-'9' or 'A' - 'F'  
 2492 (uppercase only). Most significant octet first.  
 2493 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
 2494 assigned by the client.  
 2495 This format is reserved for clients.  
 2496

2497 '3' Client URL

2498 octets 2-40: The last 39 bytes of the client URL [URI-spec].  
 2499 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
 2500 assigned by the client.  
 2501 This format is reserved for clients.  
 2502

2503 '4' Job URI

2504 octets 2-40: The last 39 bytes of the URI [URI-spec] assigned  
 2505 by the server or device to the job when the job was  
 2506 submitted for processing.  
 2507 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
 2508 assigned by the agent.  
 2509 This format is reserved for agents.  
 2510

2511 '5' POSIX User Number

2512 octets 2-40: The last 39 bytes of a user number, such as POSIX  
2513 user number.  
2514 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
2515 assigned by the client.  
2516 This format is reserved for clients.  
2517  
2518 '6' User Account Number  
2519 octets 2-40: The last 39 bytes of the user account number.  
2520 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
2521 assigned by the client.  
2522 This format is reserved for clients.  
2523  
2524 '7' DTMF Incoming FAX routing number  
2525 octets 2-40: The last 39 bytes of the DTMF incoming FAX  
2526 routing number.  
2527 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
2528 assigned by the client.  
2529 This format is reserved for clients.  
2530  
2531 '8' Job Owner supplied by the client  
2532 octets 2-40: The last 39 bytes of the job owner name (that the  
2533 agent returns in the jmJobOwner object).  
2534 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
2535 assigned by the client.  
2536 This format is reserved for clients. See format '0' which is  
2537 reserved for agents.  
2538  
2539 '9' Host Name  
2540 octets 2-40: The last 39 bytes of the host name with trailing  
2541 SPACES that submitted the job to this server/device using a  
2542 protocol, such as LPD [RFC-1179] which includes the host  
2543 name in the job submission protocol.  
2544 octets 41-48: The US-ASCII 8-decimal-digit leading zero  
2545 representation of the job id generated by the submitting  
2546 server (configuration 3) or the client (configuration 1 and  
2547 2), such as in the LPD protocol.  
2548 This format is reserved for clients.  
2549  
2550 'A' AppleTalk Protocol  
2551 octets 2-40: Contains the AppleTalk printer name, with the  
2552 first character of the name in octet 2. AppleTalk printer  
2553 names are a maximum of 31 characters. Any unused portion  
2554 of this field shall be filled with spaces.  
2555 octets 41-48: '00000XXX', where 'XXX' is the 3-digit US-ASCII  
2556 decimal representation of the Connection Id.  
2557 This format is reserved for agents.  
2558



2559            'B' NetWare PServer  
2560            octets 2-40: Contains the Directory Path Name as recorded by  
2561                    the Novell File Server in the queue directory. If the  
2562                    string is less than 40 octets, the left-most character in  
2563                    the string shall appear in octet position 2. Otherwise,  
2564                    only the last 39 bytes shall be included. Any unused  
2565                    portion of this field shall be filled with spaces.  
2566            octets 41-48: '000XXXXX' The US-ASCII representation of the  
2567                    Job Number as per the NetWare File Server Queue Management  
2568                    Services.  
2569            This format is reserved for agents.  
2570  
2571            'C' Server Message Block protocol (SMB)  
2572            octets 2-40: Contains a decimal (US-ASCII coded)  
2573                    representation of the 16 bit SMB Tree Id field, which  
2574                    uniquely identifies the connection that submitted the job  
2575                    to the printer. The most significant digit of the numeric  
2576                    string shall be placed in octet position 2. All unused  
2577                    portions of this field shall be filled with spaces. The  
2578                    SMB Tree Id has a maximum value of 65,535.  
2579            octets 41-48: The US-ASCII 8-decimal-digit leading zero  
2580                    representation of the File Handle returned from the device  
2581                    to the client in response to a Create Print File command.  
2582            This format is reserved for agents.  
2583  
2584            'D' Transport Independent Printer/System Interface (TIP/SI)  
2585            octets 2-40: Contains the Job Name from the Job Control-Start  
2586                    Job (JC-SJ) command. If the Job Name portion is less than  
2587                    40 octets, the left-most character in the string shall  
2588                    appear in octet position 2. Any unused portion of this  
2589                    field shall be filled with spaces. Otherwise, only the  
2590                    last 39 bytes shall be included.  
2591            octets 41-48: The US-ASCII 8-decimal-digit leading zero  
2592                    representation of the jmJobIndex assigned by the agent.  
2593            This format is reserved for agents, since the agent supplies  
2594                    octets 41-48, though the client supplies the job name. See  
2595                    format '1' reserved to clients to submit job name ids in  
2596                    which they supply octets 41-48.  
2597  
2598            'E' IPDS on the MVS or VSE platform  
2599  
2600            octets 2-40: Contains bytes 2-27 of the XOH Define Group  
2601                    Boundary Group ID triplet. Octet position 2 MUST carry the  
2602                    value x'01'. Bytes 28-40 MUST be filled with spaces.  
2603            octets 41-48: The US-ASCII 8-decimal-digit leading zero  
2604                    representation of the jmJobIndex assigned by the agent.  
2605            This format is reserved for agents, since the agent supplies  
2606                    octets 41-48, though the client supplies the job name.  
2607

2608 'F' IPDS on the VM platform  
2609 octets 2-40: Contains bytes 2-31 of the XOH Define Group  
2610 Boundary Group ID triplet. Octet position 2 MUST carry the  
2611 value x'02'. Bytes 32-40 MUST be filled with spaces.  
2612 octets 41-48: The US-ASCII 8-decimal-digit leading zero  
2613 representation of the jmJobIndex assigned by the agent.  
2614 This format is reserved for agents, since the agent supplies  
2615 octets 41-48, though the client supplies the file name.  
2616  
2617 'G' IPDS on the OS/400 platform  
2618 octets 2-40: Contains bytes 2-36 of the XOH Define Group  
2619 Boundary Group ID triplet. Octet position 2 MUST carry the  
2620 value x'03'. Bytes 37-40 MUST be filled with spaces.  
2621 octets 41-48: The US-ASCII 8-decimal-digit leading zero  
2622 representation of the jmJobIndex assigned by the agent.  
2623 This format is reserved for agents, since the agent supplies  
2624 octets 41-48, though the client supplies the job name.  
2625  
2626 NOTE - the job submission id is only intended to be unique  
2627 between a limited set of clients for a limited duration of  
2628 time, namely, for the life time of the job in the context of  
2629 the server or device that is processing the job. Some of the  
2630 formats include something that is unique per client and a  
2631 random number so that the same job submitted by the same client  
2632 will have a different job submission id. For other formats,  
2633 where part of the id is guaranteed to be unique for each  
2634 client, such as the MAC address or URL, a sequential number  
2635 SHOULD suffice for each client (and may be easier for each  
2636 client to manage). Therefore, the length of the job submission  
2637 id has been selected to reduce the probability of collision to  
2638 an extremely low number, but is not intended to be an absolute  
2639 guarantee of uniqueness. None-the-less, collisions are  
2640 remotely possible, but without bad consequences, since this MIB  
2641 is intended to be used only for monitoring jobs, not for  
2642 controlling and managing them.  
2643  
2644 This is like a type 2 enumeration. See section 3.7.3."  
2645 SYNTAX OCTET STRING(SIZE(1)) -- ASCII '0'-'9', 'A'-'Z', 'a'-'z'

```

2646
2647 JmJobStateTC ::= TEXTUAL-CONVENTION
2648     STATUS      current
2649     DESCRIPTION
2650         "The current state of the job (pending, processing, completed,
2651         etc.).
2652
2653     The following figure shows the normal job state transitions:
2654
2655                                     +----> canceled(7)
2656                                     /
2657 +----> pending(3) -----> processing(5) -----+-----> completed(9)
2658 |           ^           |           ^           |           \
2659 --->+       |           |           |           |           +----> aborted(8)
2660 |           v           |           v           |           /
2661 +----> pendingHeld(4)  processingStopped(6) ----+
2662

```

Figure 4 - Normal Job State Transitions

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

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

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

unknown(2),

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

pending(3),

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

pendingHeld(4),

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

2697 JmJobStateReasonsMTC (N=1..4) textual convention for the  
2698 specification of each reason.  
2699  
2700 processing(5),  
2701 One or more of:  
2702  
2703 1. the job is using, or is attempting to use, one or more  
2704 purely software processes that are analyzing, creating, or  
2705 interpreting a PDL, etc.,  
2706  
2707 2. the job is using, or is attempting to use, one or more  
2708 hardware devices that are interpreting a PDL, making marks  
2709 on a medium, and/or performing finishing, such as stapling,  
2710 etc.,  
2711  
2712 OR  
2713  
2714 3. (configuration 2) the server has made the job ready for  
2715 printing, but the output device is not yet printing it,  
2716 either because the job hasn't reached the output device or  
2717 because the job is queued in the output device or some  
2718 other spooler, awaiting the output device to print it.  
2719  
2720 When the job is in the processing state, the entire job  
2721 state includes the detailed status represented in the  
2722 device MIB indicated by the hrDeviceIndex value of the  
2723 job's physicalDevice attribute, if the agent implements  
2724 such a device MIB.  
2725  
2726 Implementations MAY, though they NEED NOT, include  
2727 additional values in the job's jmJobStateReasons1 object to  
2728 indicate the progress of the job, such as adding the  
2729 jobPrinting value to indicate when the device is actually  
2730 making marks on a medium and/or the processingToStopPoint  
2731 value to indicate that the server or device is in the  
2732 process of canceling or aborting the job.  
2733  
2734 processingStopped(6),  
2735 The job has stopped while processing for any number of  
2736 reasons and will return to the processing state as soon as  
2737 the reasons are no longer present.  
2738  
2739 The job's jmJobStateReasons1 object and/or the job's  
2740 jobStateReasonsN (N=2..4) attributes MAY indicate why the  
2741 job has stopped processing. For example, if the output  
2742 device is stopped, the deviceStopped value MAY be included  
2743 in the job's jmJobStateReasons1 object.  
2744  
2745 NOTE - When an output device is stopped, the device usually  
2746 indicates its condition in human readable form at the  
2747 device. The management application can obtain more  
2748 complete device status remotely by querying the appropriate

2749 device MIB using the job's deviceIndex attribute(s), if the  
2750 agent implements such a device MIB  
2751  
2752 canceled(7),  
2753 A client has canceled the job and the server or device has  
2754 completed canceling the job AND all MIB objects and  
2755 attributes have reached their final values for the job.  
2756 While the server or device is canceling the job, the job's  
2757 jmJobStateReasons1 object SHOULD contain the  
2758 processingToStopPoint value and one of the canceledByUser,  
2759 canceledByOperator, or canceledAtDevice values. The  
2760 canceledByUser, canceledByOperator, or canceledAtDevice  
2761 values remain while the job is in the canceled state.  
2762  
2763 aborted(8),  
2764 The job has been aborted by the system, usually while the  
2765 job was in the processing or processingStopped state and  
2766 the server or device has completed aborting the job AND all  
2767 MIB objects and attributes have reached their final values  
2768 for the job. While the server or device is aborting the  
2769 job, the job's jmJobStateReasons1 object MAY contain the  
2770 processingToStopPoint and abortedBySystem values. If  
2771 implemented, the abortedBySystem value SHALL remain while  
2772 the job is in the aborted state.  
2773  
2774 completed(9)  
2775 The job has completed successfully or with warnings or  
2776 errors after processing and all of the media have been  
2777 successfully stacked in the appropriate output bin(s) AND  
2778 all MIB objects and attributes have reached their final  
2779 values for the job. The job's jmJobStateReasons1 object  
2780 SHOULD contain one of: completedSuccessfully,  
2781 completedWithWarnings, or completedWithErrors values.  
2782  
2783 This is a type 2 enumeration. See Section 3.7.1.2."  
2784 SYNTAX INTEGER {  
2785 unknown(2),  
2786 pending(3),  
2787 pendingHeld(4),  
2788 processing(5),  
2789 processingStopped(6),  
2790 canceled(7),  
2791 aborted(8),  
2792 completed(9)  
2793 }

```
2794
2795 JmAttributeTypeTC ::= TEXTUAL-CONVENTION
2796     STATUS      current
2797     DESCRIPTION
2798         "The type of the attribute which identifies the attribute.
2799
2800     NOTE - The enum assignments are grouped logically with values
2801     assigned in groups of 20, so that additional values may be
2802     registered in the future and assigned a value that is part of
2803     their logical grouping.
2804
2805     Values in the range 2**30 to 2**31-1 are reserved for private
2806     or experimental usage. This range corresponds to the same
2807     range reserved in IPP. Implementers are warned that use of
2808     such values may conflict with other implementations.
2809     Implementers are encouraged to request registration of enum
2810     values following the procedures in Section 3.7.1.
2811
2812     See Section 3.2 entitled 'The Attribute Mechanism' for a
2813     description of this textual-convention and its use in the
2814     jmAttributeTable. See Section 3.3.8 for the specification of
2815     each attribute. The comment(s) after each enum assignment
2816     specifies the data type(s) of the attribute.
2817
2818     This is a type 2 enumeration. See Section 3.7.1.2."
2819
2820     SYNTAX      INTEGER {
2821         other(1),                -- Integer32 (-2..2147483647)
2822                                 -- AND/OR
2823                                 -- OCTET STRING(SIZE(0..63))
2824
2825         -- Job State attributes:
2826         jobStateReasons2(3),     -- JmJobStateReasons2TC
2827         jobStateReasons3(4),     -- JmJobStateReasons3TC
2828         jobStateReasons4(5),     -- JmJobStateReasons4TC
2829         processingMessage(6),    -- JmUTF8StringTC (SIZE(0..63))
2830         processingMessageNaturalLangTag(7),
2831                                 -- OCTET STRING(SIZE(0..63))
2832         jobCodedCharSet(8),      -- CodedCharSet
2833         jobNaturalLanguageTag(9), -- OCTET STRING(SIZE(0..63))
2834
```

```

2835     -- Job Identification attributes:
2836     jobURI(20),                -- OCTET STRING(SIZE(0..63))
2837     jobAccountName(21),        -- OCTET STRING(SIZE(0..63))
2838     serverAssignedJobName(22), -- JmJobStringTC (SIZE(0..63))
2839     jobName(23),               -- JmJobStringTC (SIZE(0..63))
2840     jobServiceTypes(24),       -- JmJobServiceTypesTC
2841     jobSourceChannelIndex(25), -- Integer32 (0..2147483647)
2842     jobSourcePlatformType(26), -- JmJobSourcePlatformTypeTC
2843     submittingServerName(27),  -- JmJobStringTC (SIZE(0..63))
2844     submittingApplicationName(28), -- JmJobStringTC (SIZE(0..63))
2845     jobOriginatingHost(29),    -- JmJobStringTC (SIZE(0..63))
2846     deviceNameRequested(30),   -- JmJobStringTC (SIZE(0..63))
2847     queueNameRequested(31),    -- JmJobStringTC (SIZE(0..63))
2848     physicalDevice(32),        -- hrDeviceIndex
2849                                -- AND/OR
2850                                -- JmUTF8StringTC (SIZE(0..63))
2851     numberOfDocuments(33),     -- Integer32 (-2..2147483647)
2852     fileName(34),             -- JmJobStringTC (SIZE(0..63))
2853     documentName(35),         -- JmJobStringTC (SIZE(0..63))
2854     jobComment(36),           -- JmJobStringTC (SIZE(0..63))
2855     documentFormatIndex(37),  -- Integer32 (0..2147483647)
2856     documentFormat(38),       -- PrtInterpreterLangFamilyTC
2857                                -- AND/OR
2858                                -- OCTET STRING(SIZE(0..63))
2859
2860     -- Job Parameter attributes:
2861     jobPriority(50),           -- Integer32 (-2..100)
2862     jobProcessAfterDateAndTime(51), -- DateAndTime (SNMPv2-TC)
2863     jobHold(52),              -- JmBooleanTC
2864     jobHoldUntil(53),         -- JmJobStringTC (SIZE(0..63))
2865     outputBin(54),            -- Integer32 (0..2147483647)
2866                                -- AND/OR
2867                                -- JmJobStringTC (SIZE(0..63))
2868     sides(55),                -- Integer32 (-2..2)
2869     finishing(56),            -- JmFinishingTC
2870
2871     -- Image Quality attributes:
2872     printQualityRequested(70),  -- JmPrintQualityTC
2873     printQualityUsed(71),       -- JmPrintQualityTC
2874     printerResolutionRequested(72), -- JmPrinterResolutionTC
2875     printerResolutionUsed(73),   -- JmPrinterResolutionTC
2876     tonerEcomonyRequested(74),   -- JmTonerEconomyTC
2877     tonerEcomonyUsed(75),        -- JmTonerEconomyTC
2878     tonerDensityRequested(76),   -- Integer32 (-2..100)
2879     tonerDensityUsed(77),        -- Integer32 (-2..100)
2880

```

```

2881      -- Job Progress attributes:
2882      jobCopiesRequested(90),          -- Integer32 (-2..2147483647)
2883      jobCopiesCompleted(91),         -- Integer32 (-2..2147483647)
2884      documentCopiesRequested(92),    -- Integer32 (-2..2147483647)
2885      documentCopiesCompleted(93),    -- Integer32 (-2..2147483647)
2886      jobKOctetsTransferred(94),      -- Integer32 (-2..2147483647)
2887      sheetCompletedCopyNumber(95),   -- Integer32 (-2..2147483647)
2888      sheetCompletedDocumentNumber(96),
2889                                          -- Integer32 (-2..2147483647)
2890      jobCollationType(97),           -- JmJobCollationTypeTC
2891
2892      -- Impression attributes:
2893      impressionsSpooled(110),         -- Integer32 (-2..2147483647)
2894      impressionsSentToDevice(111),    -- Integer32 (-2..2147483647)
2895      impressionsInterpreted(112),     -- Integer32 (-2..2147483647)
2896      impressionsCompletedCurrentCopy(113),
2897                                          -- Integer32 (-2..2147483647)
2898      fullColorImpressionsCompleted(114),
2899                                          -- Integer32 (-2..2147483647)
2900      highlightColorImpressionsCompleted(115),
2901                                          -- Integer32 (-2..2147483647)
2902
2903      -- Page attributes:
2904      pagesRequested(130),             -- Integer32 (-2..2147483647)
2905      pagesCompleted(131),            -- Integer32 (-2..2147483647)
2906      pagesCompletedCurrentCopy(132), -- Integer32 (-2..2147483647)
2907
2908      -- Sheet attributes:
2909      sheetsRequested(150),            -- Integer32 (-2..2147483647)
2910      sheetsCompleted(151),           -- Integer32 (-2..2147483647)
2911      sheetsCompletedCurrentCopy(152), -- Integer32 (-2..2147483647)
2912
2913      -- Resource attributes:
2914      mediumRequested(170),            -- JmMediumTypeTC
2915                                          -- AND/OR
2916                                          -- JmJobStringTC (SIZE(0..63))
2917      mediumConsumed(171),            -- Integer32 (-2..2147483647)
2918                                          -- AND
2919                                          -- JmJobStringTC (SIZE(0..63))
2920      colorantRequested(172),         -- Integer32 (-2..2147483647)
2921                                          -- AND/OR
2922                                          -- JmJobStringTC (SIZE(0..63))
2923      colorantConsumed(173),         -- Integer32 (-2..2147483647)
2924                                          -- AND/OR
2925                                          -- JmJobStringTC (SIZE(0..63))
2926      mediumTypeConsumed(174),        -- Integer32 (-2..2147483647)
2927                                          -- AND
2928                                          -- JmJobStringTC (SIZE(0..63))
2929      mediumSizeConsumed(175),        -- Integer32 (-2..2147483647)
2930                                          -- AND
2931                                          -- JmJobStringTC (SIZE(0..63))
2932

```



```
2933      -- Time attributes:
2934      jobSubmissionToServerTime(190), -- JmTimeStampTC
2935                                          -- AND/OR
2936                                          -- DateAndTime
2937      jobSubmissionTime(191),          -- JmTimeStampTC
2938                                          -- AND/OR
2939                                          -- DateAndTime
2940      jobStartedBeingHeldTime(192),    -- JmTimeStampTC
2941                                          -- AND/OR
2942                                          -- DateAndTime
2943      jobStartedProcessingTime(193),   -- JmTimeStampTC
2944                                          -- AND/OR
2945                                          -- DateAndTime
2946      jobCompletionTime(194),          -- JmTimeStampTC
2947                                          -- AND/OR
2948                                          -- DateAndTime
2949      jobProcessingCPUTime(195)        -- Integer32 (-2..2147483647)
2950  }
2951
```

2952 JmJobServiceTypesTC ::= TEXTUAL-CONVENTION  
2953     STATUS         current  
2954     DESCRIPTION  
2955         "Specifies the type(s) of service to which the job has been  
2956         submitted (print, fax, scan, etc.). The service type is  
2957         represented as an enum that is bit encoded with each job  
2958         service type so that more general and arbitrary services can be  
2959         created, such as services with more than one destination type,  
2960         or ones with only a source or only a destination. For example,  
2961         a job service might scan, faxOut, and print a single job. In  
2962         this case, three bits would be set in the jobServiceTypes  
2963         attribute, corresponding to the hexadecimal values: 0x8 + 0x20  
2964         + 0x4, respectively, yielding: 0x2C.  
2965  
2966         Whether this attribute is set from a job attribute supplied by  
2967         the job submission client or is set by the recipient job  
2968         submission server or device depends on the job submission  
2969         protocol. With either implementation, the agent SHALL return a  
2970         non-zero value for this attribute indicating the type of the  
2971         job.  
2972  
2973         One of the purposes of this attribute is to permit a requester  
2974         to filter out jobs that are not of interest. For example, a  
2975         printer operator MAY only be interested in jobs that include  
2976         printing. That is why the attribute is in the job  
2977         identification category.  
2978  
2979         The following service component types are defined (in  
2980         hexadecimal) and are assigned a separate bit value for use with  
2981         the jobServiceTypes attribute:  
2982  
2983         other                                 0x1  
2984             The job contains some instructions that are not one of the  
2985             identified types.  
2986  
2987         unknown                               0x2  
2988             The job contains some instructions whose type is unknown to  
2989             the agent.  
2990  
2991         print                                 0x4  
2992             The job contains some instructions that specify printing  
2993  
2994         scan                                 0x8  
2995             The job contains some instructions that specify scanning  
2996  
2997         faxIn                                 0x10  
2998             The job contains some instructions that specify receive fax  
2999  
3000         faxOut                               0x20  
3001             The job contains some instructions that specify sending fax  
3002



3055  
3056 other 0x1  
3057 The job state reason is not one of the standardized or  
3058 registered reasons.  
3059  
3060 unknown 0x2  
3061 The job state reason is not known to the agent or is  
3062 indeterminent.  
3063  
3064 jobIncoming 0x4  
3065 The job has been accepted by the server or device, but the  
3066 server or device is expecting (1) additional operations  
3067 from the client to finish creating the job and/or (2) is  
3068 accessing/accepting document data.  
3069  
3070 submissionInterrupted 0x8  
3071 The job was not completely submitted for some unforeseen  
3072 reason, such as: (1) the server has crashed before the job  
3073 was closed by the client, (2) the server or the document  
3074 transfer method has crashed in some non-recoverable way  
3075 before the document data was entirely transferred to the  
3076 server, (3) the client crashed or failed to close the job  
3077 before the time-out period.  
3078  
3079 jobOutgoing 0x10  
3080 Configuration 2 only: The server is transmitting the job  
3081 to the device.  
3082  
3083 jobHoldSpecified 0x20  
3084 The value of the job's jobHold(52) attribute is TRUE. The  
3085 job SHALL NOT be a candidate for processing until this  
3086 reason is removed and there are no other reasons to hold  
3087 the job.  
3088  
3089 jobHoldUntilSpecified 0x40  
3090 The value of the job's jobHoldUntil(53) attribute specifies  
3091 a time period that is still in the future. The job SHALL  
3092 NOT be a candidate for processing until this reason is  
3093 removed and there are no other reasons to hold the job.  
3094  
3095 jobProcessAfterSpecified 0x80  
3096 The value of the job's jobProcessAfterDateAndTime(51)  
3097 attribute specifies a time that is still in the future.  
3098 The job SHALL NOT be a candidate for processing until this  
3099 reason is removed and there are no other reasons to hold  
3100 the job.  
3101

3102 resourcesAreNotReady 0x100  
3103 At least one of the resources needed by the job, such as  
3104 media, fonts, resource objects, etc., is not ready on any  
3105 of the physical devices for which the job is a candidate.  
3106 This condition MAY be detected when the job is accepted, or  
3107 subsequently while the job is pending or processing,  
3108 depending on implementation.  
3109

3110 deviceStoppedPartly 0x200  
3111 One or more, but not all, of the devices to which the job  
3112 is assigned are stopped. If all of the devices are stopped  
3113 (or the only device is stopped), the deviceStopped reason  
3114 SHALL be used.  
3115

3116 deviceStopped 0x400  
3117 The device(s) to which the job is assigned is (are all)  
3118 stopped.  
3119

3120 jobInterpreting 0x800  
3121 The device to which the job is assigned is interpreting the  
3122 document data.  
3123

3124 jobPrinting 0x1000  
3125 The output device to which the job is assigned is marking  
3126 media. This value is useful for servers and output devices  
3127 which spend a great deal of time processing (1) when no  
3128 marking is happening and then want to show that marking is  
3129 now happening or (2) when the job is in the process of  
3130 being canceled or aborted while the job remains in the  
3131 processing state, but the marking has not yet stopped so  
3132 that impression or sheet counts are still increasing for  
3133 the job.  
3134

3135 jobCanceledByUser 0x2000  
3136 The job was canceled by the owner of the job, i.e., by a  
3137 user whose name is the same as the value of the job's  
3138 jmJobOwner object, or by some other authorized end-user,  
3139 such as a member of the job owner's security group.  
3140

3141 jobCanceledByOperator 0x4000  
3142 The job was canceled by the operator, i.e., by a user who  
3143 has been authenticated as having operator privileges  
3144 (whether local or remote).  
3145

3146 jobCanceledAtDevice 0x8000  
3147 The job was canceled by an unidentified local user, i.e., a  
3148 user at a console at the device.  
3149

3150 abortedBySystem 0x10000  
3151 The job (1) is in the process of being aborted, (2) has  
3152 been aborted by the system and placed in the 'aborted'  
3153 state, or (3) has been aborted by the system and placed in  
3154 the 'pendingHeld' state, so that a user or operator can  
3155 manually try the job again.  
3156

3157 processingToStopPoint 0x20000  
3158 The requester has issued an operation to cancel or  
3159 interrupt the job or the server/device has aborted the job,  
3160 but the server/device is still performing some actions on  
3161 the job until a specified stop point occurs or job  
3162 termination/cleanup is completed.  
3163

3164 This reason is recommended to be used in conjunction with  
3165 the processing job state to indicate that the server/device  
3166 is still performing some actions on the job while the job  
3167 remains in the processing state. After all the job's  
3168 resources consumed counters have stopped incrementing, the  
3169 server/device moves the job from the processing state to  
3170 the canceled or aborted job states.  
3171

3172 serviceOffLine 0x40000  
3173 The service or document transform is off-line and accepting  
3174 no jobs. All pending jobs are put into the pendingHeld  
3175 state. This situation could be true if the service's or  
3176 document transform's input is impaired or broken.  
3177

3178 jobCompletedSuccessfully 0x80000  
3179 The job completed successfully.  
3180

3181 jobCompletedWithWarnings 0x100000  
3182 The job completed with warnings.  
3183

3184 jobCompletedWithErrors 0x200000  
3185 The job completed with errors (and possibly warnings too).  
3186  
3187

3188 The following additional job state reasons have been added to  
3189 represent job states that are in ISO DPA[iso-dpa] and other job  
3190 submission protocols:  
3191

3192 jobPaused 0x400000  
3193 The job has been indefinitely suspended by a client issuing  
3194 an operation to suspend the job so that other jobs may  
3195 proceed using the same devices. The client MAY issue an  
3196 operation to resume the paused job at any time, in which  
3197 case the agent SHALL remove the jobPaused values from the  
3198 job's jmJobStateReasons1 object and the job is eventually  
3199 resumed at or near the point where the job was paused.  
3200

3201           jobInterrupted                           0x800000  
3202           The job has been interrupted while processing by a client  
3203           issuing an operation that specifies another job to be run  
3204           instead of the current job. The server or device will  
3205           automatically resume the interrupted job when the  
3206           interrupting job completes.  
3207  
3208           jobRetained                               0x1000000  
3209           The job is being retained by the server or device with all  
3210           of the job's document data (and submitted resources, such  
3211           as fonts, logos, and forms, if any). Thus a client could  
3212           issue an operation to the server or device to either (1)  
3213           re-do the job (or a copy of the job) on the same server or  
3214           device or (2) resubmit the job to another server or device.  
3215           When a client could no longer re-do/resubmit the job, such  
3216           as after the document data has been discarded, the agent  
3217           SHALL remove the jobRetained value from the  
3218           jmJobStateReasons1 object.  
3219  
3220           These bit definitions are the equivalent of a type 2 enum  
3221           except that combinations of bits may be used together. See  
3222           section 3.7.1.2. The remaining bits are reserved for future  
3223           standardization and/or registration."  
3224           SYNTAX            INTEGER (0..2147483647)    -- 31 bits, all but sign bit  
3225  
3226  
3227  
3228           JmJobStateReasons2TC ::= TEXTUAL-CONVENTION  
3229           STATUS            current  
3230           DESCRIPTION  
3231            "This textual-convention is used with the jobStateReasons2  
3232            attribute to provides additional information regarding the  
3233            jmJobState object. See the description under  
3234            JmJobStateReasons1TC for additional information that applies to  
3235            all reasons.  
3236  
3237            The following standard values are defined (in hexadecimal) as  
3238            *powers of two*, since multiple values may be used at the same  
3239            time:  
3240  
3241            cascaded                                    0x1  
3242            An outbound gateway has transmitted all of the job's job  
3243            and document attributes and data to another spooling  
3244            system.  
3245  
3246            deletedByAdministrator                    0x2  
3247            The administrator has deleted the job.  
3248  
3249            discardTimeArrived                         0x4  
3250            The job has been deleted due to the fact that the time  
3251            specified by the job's job-discard-time attribute has  
3252            arrived.

3253  
3254       postProcessingFailed                   0x8  
3255       The post-processing agent failed while trying to log  
3256       accounting attributes for the job; therefore the job has  
3257       been placed into the completed state with the jobRetained  
3258       jmJobStateReasons1 object value for a system-defined period  
3259       of time, so the administrator can examine it, resubmit it,  
3260       etc.  
3261  
3262       jobTransforming                        0x10  
3263       The server/device is interpreting document data and  
3264       producing another electronic representation.  
3265  
3266       maxJobFaultCountExceeded               0x20  
3267       The job has faulted several times and has exceeded the  
3268       administratively defined fault count limit.  
3269  
3270       devicesNeedAttentionTimeOut            0x40  
3271       One or more document transforms that the job is using needs  
3272       human intervention in order for the job to make progress,  
3273       but the human intervention did not occur within the site-  
3274       settable time-out value.  
3275  
3276       needsKeyOperatorTimeOut                0x80  
3277       One or more devices or document transforms that the job is  
3278       using need a specially trained operator (who may need a key  
3279       to unlock the device and gain access) in order for the job  
3280       to make progress, but the key operator intervention did not  
3281       occur within the site-settable time-out value.  
3282  
3283       jobStartWaitTimeOut                    0x100  
3284       The server/device has stopped the job at the beginning of  
3285       processing to await human action, such as installing a  
3286       special cartridge or special non-standard media, but the  
3287       job was not resumed within the site-settable time-out value  
3288       and the server/device has transitioned the job to the  
3289       pendingHeld state.  
3290  
3291       jobEndWaitTimeOut                      0x200  
3292       The server/device has stopped the job at the end of  
3293       processing to await human action, such as removing a  
3294       special cartridge or restoring standard media, but the job  
3295       was not resumed within the site-settable time-out value and  
3296       the server/device has transitioned the job to the completed  
3297       state.  
3298  
3299       jobPasswordWaitTimeOut                 0x400  
3300       The server/device has stopped the job at the beginning of  
3301       processing to await input of the job's password, but the  
3302       password was not received within the site-settable time-out  
3303       value.  
3304



3305 deviceTimedOut 0x800  
3306 A device that the job was using has not responded in a  
3307 period specified by the device's site-settable attribute.  
3308  
3309 connectingToDeviceTimeOut 0x1000  
3310 The server is attempting to connect to one or more devices  
3311 which may be dial-up, polled, or queued, and so may be busy  
3312 with traffic from other systems, but server was unable to  
3313 connect to the device within the site-settable time-out  
3314 value.  
3315  
3316 transferring 0x2000  
3317 The job is being transferred to a down stream server or  
3318 downstream device.  
3319  
3320 queuedInDevice 0x4000  
3321 The server/device has queued the job in a down stream  
3322 server or downstream device.  
3323  
3324 jobQueued 0x8000  
3325 The server/device has queued the document data.  
3326  
3327 jobCleanup 0x10000  
3328 The server/device is performing cleanup activity as part of  
3329 ending normal processing.  
3330  
3331 jobPasswordWait 0x20000  
3332 The server/device has selected the job to be next to  
3333 process, but instead of assigning resources and starting  
3334 the job processing, the server/device has transitioned the  
3335 job to the pendingHeld state to await entry of a password  
3336 (and dispatched another job, if there is one).  
3337  
3338 validating 0x40000  
3339 The server/device is validating the job *after* accepting the  
3340 job.  
3341  
3342 queueHeld 0x80000  
3343 The operator has held the entire job set or queue.  
3344  
3345 jobProofWait 0x100000  
3346 The job has produced a single proof copy and is in the  
3347 pendingHeld state waiting for the requester to issue an  
3348 operation to release the job to print normally, obeying any  
3349 job and document copy attributes that were originally  
3350 submitted.  
3351  
3352 heldForDiagnostics 0x200000  
3353 The system is running intrusive diagnostics, so that all  
3354 jobs are being held.

3355 noSpaceOnServer 0x800000  
3356 There is no room on the server to store all of the job.  
3357  
3358 pinRequired 0x1000000  
3359 The System Administrator settable device policy is (1) to  
3360 require PINs, and (2) to hold jobs that do not have a pin  
3361 supplied as an input parameter when the job was created.  
3362  
3363 exceededAccountLimit 0x2000000  
3364 The account for which this job is drawn has exceeded its  
3365 limit. This condition SHOULD be detected before the job is  
3366 scheduled so that the user does not wait until his/her job  
3367 is scheduled only to find that the account is overdrawn.  
3368 This condition MAY also occur while the job is processing  
3369 either as processing begins or part way through processing.  
3370  
3371 heldForRetry 0x4000000  
3372 The job encountered some errors that the server/device  
3373 could not recover from with its normal retry procedures,  
3374 but the error might not be encountered if the job is  
3375 processed again in the future. Example cases are phone  
3376 number busy or remote file system in-accessible. For such  
3377 a situation, the server/device SHALL transition the job  
3378 from the processing to the pendingHeld, rather than to the  
3379 aborted state.  
3380  
3381 The following values are from the X/Open PSIS draft standard:  
3382  
3383 canceledByShutdown 0x8000000  
3384 The job was canceled because the server or device was  
3385 shutdown before completing the job.  
3386  
3387 deviceUnavailable 0x10000000  
3388 This job was aborted by the system because the device is  
3389 currently unable to accept jobs.  
3390  
3391 wrongDevice 0x20000000  
3392 This job was aborted by the system because the device is  
3393 unable to handle this particular job; the spooler SHOULD  
3394 try another device or the user should submit the job to  
3395 another device.  
3396  
3397 badJob 0x40000000  
3398 This job was aborted by the system because this job has a  
3399 major problem, such as an ill-formed PDL; the spooler  
3400 SHOULD not even try another device.  
3401  
3402 These bit definitions are the equivalent of a type 2 enum  
3403 except that combinations of them may be used together. See  
3404 section 3.7.1.2. See the description under  
3405 JmJobStateReasons1TC and the jobStateReasons2 attribute."  
3406 SYNTAX INTEGER (0..2147483647) -- 31 bits, all but sign bit

3407  
3408 JmJobStateReasons3TC ::= TEXTUAL-CONVENTION  
3409     STATUS         current  
3410     DESCRIPTION  
3411         "This textual-convention is used with the jobStateReasons3  
3412         attribute to provides additional information regarding the  
3413         jmJobState object. See the description under  
3414         JmJobStateReasons1TC for additional information that applies to  
3415         all reasons.  
3416  
3417         The following standard values are defined (in hexadecimal) as  
3418         *powers of two*, since multiple values may be used at the same  
3419         time:  
3420  
3421         jobInterruptedByDeviceFailure         0x1  
3422             A device or the print system software that the job was  
3423             using has failed while the job was processing. The server  
3424             or device is keeping the job in the pendingHeld state until  
3425             an operator can determine what to do with the job.  
3426  
3427         These bit definitions are the equivalent of a type 2 enum  
3428         except that combinations of them may be used together. See  
3429         section 3.7.1.2. The remaining bits are reserved for future  
3430         standardization and/or registration. See the description under  
3431         JmJobStateReasons1TC and the jobStateReasons3 attribute."  
3432     SYNTAX         INTEGER (0..2147483647)     -- 31 bits, all but sign bit  
3433  
3434  
3435  
3436  
3437  
3438 JmJobStateReasons4TC ::= TEXTUAL-CONVENTION  
3439     STATUS         current  
3440     DESCRIPTION  
3441         "This textual-convention is used in the jobStateReasons4  
3442         attribute to provides additional information regarding the  
3443         jmJobState object. See the description under  
3444         JmJobStateReasons1TC for additional information that applies to  
3445         all reasons.  
3446  
3447         The following standard values are defined (in hexadecimal) as  
3448         *powers of two*, since multiple values may be used at the same  
3449         time:  
3450  
3451         none yet defined. These bits are reserved for future  
3452         standardization and/or registration.  
3453  
3454         These bit definitions are the equivalent of a type 2 enum  
3455         except that combinations of them may be used together. See  
3456         section 3.7.1.2. See the description under  
3457         JmJobStateReasons1TC and the jobStateReasons4 attribute."  
3458     SYNTAX         INTEGER (0..2147483647)     -- 31 bits, all but sign bit

```

3459
3460 jobmonMIBObjects OBJECT IDENTIFIER ::= { jobmonMIB 1 }
3461
3462 -- The General Group (MANDATORY)
3463
3464 -- The jmGeneralGroup consists entirely of the jmGeneralTable.
3465
3466 jmGeneral OBJECT IDENTIFIER ::= { jobmonMIBObjects 1 }
3467
3468 jmGeneralTable OBJECT-TYPE
3469     SYNTAX      SEQUENCE OF JmGeneralEntry
3470     MAX-ACCESS  not-accessible
3471     STATUS      current
3472     DESCRIPTION
3473         "The jmGeneralTable consists of information of a general nature
3474         that are per-job-set, but are not per-job. See Section 2
3475         entitled 'Terminology and Job Model' for the definition of a
3476         job set.
3477
3478         The MANDATORY-GROUP macro specifies that this group is
3479         MANDATORY."
3480     ::= { jmGeneral 1 }
3481
3482
3483 jmGeneralEntry OBJECT-TYPE
3484     SYNTAX      JmGeneralEntry
3485     MAX-ACCESS  not-accessible
3486     STATUS      current
3487     DESCRIPTION
3488         "Information about a job set (queue).
3489
3490         An entry SHALL exist in this table for each job set."
3491     INDEX      { jmGeneralJobSetIndex }
3492     ::= { jmGeneralTable 1 }
3493
3494
3495 JmGeneralEntry ::= SEQUENCE {
3496     jmGeneralJobSetIndex      Integer32 (1..32767),
3497     jmGeneralNumberOfActiveJobs Integer32 (0..2147483647),
3498     jmGeneralOldestActiveJobIndex Integer32 (0..2147483647),
3499     jmGeneralNewestActiveJobIndex Integer32 (0..2147483647),
3500     jmGeneralJobPersistence     Integer32 (15..2147483647),
3501     jmGeneralAttributePersistence Integer32 (15..2147483647),
3502     jmGeneralJobSetName         JmUTF8StringTC (SIZE(0..63))
3503 }
3504

```

```
3505 jmGeneralJobSetIndex OBJECT-TYPE
3506     SYNTAX      Integer32 (1..32767)
3507     MAX-ACCESS  not-accessible
3508     STATUS      current
3509     DESCRIPTION
3510         "A unique value for each job set in this MIB.  The jmJobTable
3511         and jmAttributeTable tables have this same index as their
3512         primary index.
3513
3514         The value(s) of the jmGeneralJobSetIndex SHALL be persistent
3515         across power cycles, so that clients that have retained
3516         jmGeneralJobSetIndex values will access the same job sets upon
3517         subsequent power-up.
3518
3519         An implementation that has only one job set, such as a printer
3520         with a single queue, SHALL hard code this object with the value
3521         1.
3522
3523         See Section 2 entitled 'Terminology and Job Model' for the
3524         definition of a job set.
3525         Corresponds to the first index in jmJobTable and
3526         jmAttributeTable."
3527     ::= { jmGeneralEntry 1 }
3528
3529
3530 jmGeneralNumberOfActiveJobs OBJECT-TYPE
3531     SYNTAX      Integer32 (0..2147483647)
3532     MAX-ACCESS  read-only
3533     STATUS      current
3534     DESCRIPTION
3535         "The current number of 'active' jobs in the jmJobIDTable,
3536         jmJobTable, and jmAttributeTable, i.e., the total number of
3537         jobs that are in the pending, processing, or processingStopped
3538         states.  See the JmJobStateTC textual-convention for the exact
3539         specification of the semantics of the job states."
3540     DEFVAL      { 0 }      -- no jobs
3541     ::= { jmGeneralEntry 2 }
3542
```

```
3543 jmGeneralOldestActiveJobIndex OBJECT-TYPE
3544     SYNTAX      Integer32 (0..2147483647)
3545     MAX-ACCESS  read-only
3546     STATUS      current
3547     DESCRIPTION
3548         "The jmJobIndex of the oldest job that is still in one of the
3549         'active' states (pending, processing, or processingStopped).
3550         In other words, the index of the 'active' job that has been in
3551         the job tables the longest.
3552
3553         If there are no active jobs, the agent SHALL set the value of
3554         this object to 0.
3555
3556         See Section 3.2 entitled 'The Job Tables and the Oldest Active
3557         and Newest Active Indexes' for a description of the usage of
3558         this object."
3559     DEFVAL      { 0 }      -- no active jobs
3560     ::= { jmGeneralEntry 3 }
3561
3562
3563
3564 jmGeneralNewestActiveJobIndex OBJECT-TYPE
3565     SYNTAX      Integer32 (0..2147483647)
3566     MAX-ACCESS  read-only
3567     STATUS      current
3568     DESCRIPTION
3569         "The jmJobIndex of the newest job that is in one of the
3570         'active' states (pending, processing, or processingStopped).
3571         In other words, the index of the 'active' job that has been
3572         most recently added to the job tables.
3573
3574         When all jobs become 'inactive', i.e., enter the pendingHeld,
3575         completed, canceled, or aborted states, the agent SHALL set the
3576         value of this object to 0.
3577
3578         See Section 3.2 entitled 'The Job Tables and the Oldest Active
3579         and Newest Active Indexes' for a description of the usage of
3580         this object."
3581     DEFVAL      { 0 }      -- no active jobs
3582     ::= { jmGeneralEntry 4 }
3583
```

```
3584 jmGeneralJobPersistence OBJECT-TYPE
3585     SYNTAX      Integer32 (15..2147483647)
3586     UNITS       "seconds"
3587     MAX-ACCESS  read-only
3588     STATUS      current
3589     DESCRIPTION
3590         "The minimum time in seconds for this instance of the Job Set
3591         that an entry SHALL remain in the jmJobIDTable and jmJobTable
3592         after processing has completed, i.e., the minimum time in
3593         seconds starting when the job enters the completed, canceled,
3594         or aborted state.
3595
3596         Configuring this object is implementation-dependent.
3597
3598         This value SHALL be equal to or greater than the value of
3599         jmGeneralAttributePersistence. This value SHOULD be at least
3600         60 which gives a monitoring or accounting application one
3601         minute in which to poll for job data."
3602     DEFVAL      { 60 }          -- one minute
3603     ::= { jmGeneralEntry 5 }
3604
3605
3606
3607 jmGeneralAttributePersistence OBJECT-TYPE
3608     SYNTAX      Integer32 (15..2147483647)
3609     UNITS       "seconds"
3610     MAX-ACCESS  read-only
3611     STATUS      current
3612     DESCRIPTION
3613         "The minimum time in seconds for this instance of the Job Set
3614         that an entry SHALL remain in the jmAttributeTable after
3615         processing has completed , i.e., the time in seconds starting
3616         when the job enters the completed, canceled, or aborted state.
3617
3618         Configuring this object is implementation-dependent.
3619
3620         This value SHOULD be at least 60 which gives a monitoring or
3621         accounting application one minute in which to poll for job
3622         data."
3623     DEFVAL      { 60 }          -- one minute
3624     ::= { jmGeneralEntry 6 }
3625
```

```
3626 jmGeneralJobSetName OBJECT-TYPE
3627     SYNTAX      JmUTF8StringTC (SIZE(0..63))
3628     MAX-ACCESS  read-only
3629     STATUS      current
3630     DESCRIPTION
3631         "The human readable name of this job set assigned by the system
3632         administrator (by means outside of this MIB).  Typically, this
3633         name SHOULD be the name of the job queue.  If a server or
3634         device has only a single job set, this object can be the
3635         administratively assigned name of the server or device itself.
3636         This name does not need to be unique, though each job set in a
3637         single Job Monitoring MIB SHOULD have distinct names.
3638
3639         NOTE - If the job set corresponds to a single printer and the
3640         Printer MIB is implemented, this value SHOULD be the same as
3641         the prtGeneralPrinterName object in the draft Printer MIB
3642         [print-mib-draft].  If the job set corresponds to an IPP
3643         Printer, this value SHOULD be the same as the IPP 'printer-
3644         name' Printer attribute.
3645
3646         NOTE - The purpose of this object is to help the user of the
3647         job monitoring application distinguish between several job sets
3648         in implementations that support more than one job set.
3649
3650         See the OBJECT compliance macro for the minimum maximum length
3651         required for conformance."
3652     DEFVAL      { ''H }      -- empty string
3653     ::= { jmGeneralEntry 7 }
```



```

3659 -- The Job ID Group (MANDATORY)
3660
3661 -- The jmJobIDGroup consists entirely of the jmJobIDTable.
3662
3663 jmJobID OBJECT IDENTIFIER ::= { jobmonMIBObjects 2 }
3664
3665 jmJobIDTable OBJECT-TYPE
3666     SYNTAX      SEQUENCE OF JmJobIDEntry
3667     MAX-ACCESS  not-accessible
3668     STATUS      current
3669     DESCRIPTION
3670         "The jmJobIDTable provides a correspondence map (1) between the
3671         job submission ID that a client uses to refer to a job and (2)
3672         the jmGeneralJobSetIndex and jmJobIndex that the Job Monitoring
3673         MIB agent assigned to the job and that are used to access the
3674         job in all of the other tables in the MIB.  If a monitoring
3675         application already knows the jmGeneralJobSetIndex and the
3676         jmJobIndex of the job it is querying, that application NEED NOT
3677         use the jmJobIDTable.
3678
3679         The MANDATORY-GROUP macro specifies that this group is
3680         MANDATORY."
3681     ::= { jmJobID 1 }
3682
3683
3684
3685 jmJobIDEntry OBJECT-TYPE
3686     SYNTAX      JmJobIDEntry
3687     MAX-ACCESS  not-accessible
3688     STATUS      current
3689     DESCRIPTION
3690         "The map from (1) the jmJobSubmissionID to (2) the
3691         jmGeneralJobSetIndex and jmJobIndex.
3692
3693         An entry SHALL exist in this table for each job currently known
3694         to the agent for all job sets and job states.  There MAY be
3695         more than one jmJobIDEntry that maps to a single job.  This
3696         many to one mapping can occur when more than one network entity
3697         along the job submission path supplies a job submission ID.
3698         See Section 3.5.  However, each job SHALL appear once and in
3699         one and only one job set."
3700     INDEX { jmJobSubmissionID }
3701     ::= { jmJobIDTable 1 }
3702
3703 JmJobIDEntry ::= SEQUENCE {
3704     jmJobSubmissionID          OCTET STRING(SIZE(48)),
3705     jmJobIDJobSetIndex         Integer32 (0..32767),
3706     jmJobIDJobIndex           Integer32 (0..2147483647)
3707 }
3708

```

3709 jmJobSubmissionID OBJECT-TYPE  
3710     SYNTAX         OCTET STRING(SIZE(48))  
3711     MAX-ACCESS     not-accessible  
3712     STATUS         current  
3713     DESCRIPTION  
3714         "A quasi-unique 48-octet fixed-length string ID which  
3715         identifies the job within a particular client-server  
3716         environment. There are multiple formats for the  
3717         jmJobSubmissionID. Each format SHALL be uniquely identified.  
3718         See the JmJobSubmissionIDTypeTC textual convention. Each  
3719         format SHALL be registered using the procedures of a type 2  
3720         enum. See section 3.7.3 entitled: 'PWG Registration of Job  
3721         Submission Id Formats'.  
3722  
3723         If the requester (client or server) does not supply a job  
3724         submission ID in the job submission protocol, then the  
3725         recipient (server or device) SHALL assign a job submission ID  
3726         using any of the standard formats that have been reserved for  
3727         agents and adding the final 8 octets to distinguish the ID from  
3728         others submitted from the same requester.  
3729  
3730         The monitoring application, whether in the client or running  
3731         separately, MAY use the job submission ID to help identify  
3732         which jmJobIndex was assigned by the agent, i.e., in which row  
3733         the job information is in the other tables.  
3734  
3735         NOTE - fixed-length is used so that a management application  
3736         can use a shortened GetNext varbind (in SNMPv1 and SNMPv2) in  
3737         order to get the next submission ID, disregarding the remainder  
3738         of the ID in order to access jobs independent of the trailing  
3739         identifier part, e.g., to get all jobs submitted by a  
3740         particular jmJobOwner or submitted from a particular MAC  
3741         address.  
3742  
3743         See the JmJobSubmissionIDTypeTC textual convention.  
3744         See APPENDIX B - Support of Job Submission Protocols."  
3745      ::= { jmJobIDEntry 1 }  
3746

```
3747 jmJobIDJobSetIndex OBJECT-TYPE
3748     SYNTAX      Integer32 (0..32767)
3749     MAX-ACCESS  read-only
3750     STATUS      current
3751     DESCRIPTION
3752         "This object contains the value of the jmGeneralJobSetIndex for
3753         the job with the jmJobSubmissionID value, i.e., the job set
3754         index of the job set in which the job was placed when that
3755         server or device accepted the job. This 16-bit value in
3756         combination with the jmJobIDJobIndex value permits the
3757         management application to access the other tables to obtain the
3758         job-specific objects for this job.
3759
3760         See jmGeneralJobSetIndex in the jmGeneralTable."
3761     DEFVAL      { 0 }      -- 0 indicates no job set index
3762     ::= { jmJobIDEntry 2 }
3763
3764
3765
3766 jmJobIDJobIndex OBJECT-TYPE
3767     SYNTAX      Integer32 (0..2147483647)
3768     MAX-ACCESS  read-only
3769     STATUS      current
3770     DESCRIPTION
3771         "This object contains the value of the jmJobIndex for the job
3772         with the jmJobSubmissionID value, i.e., the job index for the
3773         job when the server or device accepted the job. This value, in
3774         combination with the jmJobIDJobSetIndex value, permits the
3775         management application to access the other tables to obtain the
3776         job-specific objects for this job.
3777
3778         See jmJobIndex in the jmJobTable."
3779     DEFVAL      { 0 }      -- 0 indicates no jmJobIndex value.
3780     ::= { jmJobIDEntry 3 }
3781
3782
3783
3784
```

```

3785 -- The Job Group (MANDATORY)
3786
3787 -- The jmJobGroup consists entirely of the jmJobTable.
3788
3789 jmJob OBJECT IDENTIFIER ::= { jobmonMIBObjects 3 }
3790
3791 jmJobTable OBJECT-TYPE
3792     SYNTAX      SEQUENCE OF JmJobEntry
3793     MAX-ACCESS  not-accessible
3794     STATUS      current
3795     DESCRIPTION
3796         "The jmJobTable consists of basic job state and status
3797         information for each job in a job set that (1) monitoring
3798         applications need to be able to access in a single SNMP Get
3799         operation, (2) that have a single value per job, and (3) that
3800         SHALL always be implemented.
3801
3802         The MANDATORY-GROUP macro specifies that this group is
3803         MANDATORY."
3804     ::= { jmJob 1 }
3805
3806
3807
3808 jmJobEntry OBJECT-TYPE
3809     SYNTAX      JmJobEntry
3810     MAX-ACCESS  not-accessible
3811     STATUS      current
3812     DESCRIPTION
3813         "Basic per-job state and status information.
3814
3815         An entry SHALL exist in this table for each job, no matter what
3816         the state of the job is. Each job SHALL appear in one and only
3817         one job set.
3818
3819         See Section 3.2 entitled 'The Job Tables'."
3820     INDEX { jmGeneralJobSetIndex, jmJobIndex }
3821     ::= { jmJobTable 1 }
3822
3823 JmJobEntry ::= SEQUENCE {
3824     jmJobIndex          Integer32 (1..2147483647),
3825     jmJobState          JmJobStateTC,
3826     jmJobStateReasons1 JmJobStateReasons1TC,
3827     jmNumberOfInterveningJobs Integer32 (-2..2147483647),
3828     jmJobKOctetsPerCopyRequested Integer32 (-2..2147483647),
3829     jmJobKOctetsProcessed Integer32 (-2..2147483647),
3830     jmJobImpressionsPerCopyRequested Integer32 (-2..2147483647),
3831     jmJobImpressionsCompleted Integer32 (-2..2147483647),
3832     jmJobOwner          JmJobStringTC (SIZE(0..63))
3833 }
3834

```

```
3835 jmJobIndex OBJECT-TYPE
3836     SYNTAX      Integer32 (1..2147483647)
3837     MAX-ACCESS  not-accessible
3838     STATUS      current
3839     DESCRIPTION
3840         "The sequential, monotonically increasing identifier index for
3841         the job generated by the server or device when that server or
3842         device accepted the job. This index value permits the
3843         management application to access the other tables to obtain the
3844         job-specific row entries.
3845
3846         See Section 3.2 entitled 'The Job Tables and the Oldest Active
3847         and Newest Active Indexes'.
3848         See Section 3.5 entitled 'Job Identification'.
3849         See also
3850
3851         jmGeneralNewestActiveJobIndex for the largest value of
3852         jmJobIndex.
3853         See JmJobSubmissionIDTypeTC for a limit on the size of this
3854         index if the agent represents it as an 8-digit decimal number."
3855     ::= { jmJobEntry 1 }
3856
3857
3858
3859 jmJobState OBJECT-TYPE
3860     SYNTAX      JmJobStateTC
3861     MAX-ACCESS  read-only
3862     STATUS      current
3863     DESCRIPTION
3864         "The current state of the job (pending, processing, completed,
3865         etc.). Agents SHALL implement only those states which are
3866         appropriate for the particular implementation. However,
3867         management applications SHALL be prepared to receive all the
3868         standard job states.
3869
3870         The final value for this object SHALL be one of: completed,
3871         canceled, or aborted. The minimum length of time that the
3872         agent SHALL maintain MIB data for a job in the completed,
3873         canceled, or aborted state before removing the job data from
3874         the jmJobIDTable and jmJobTable is specified by the value of
3875         the jmGeneralJobPersistence object."
3876     DEFVAL      { unknown }          -- default is unknown
3877     ::= { jmJobEntry 2 }
3878
```

```
3879 jmJobStateReasons1 OBJECT-TYPE
3880     SYNTAX      JmJobStateReasons1TC
3881     MAX-ACCESS  read-only
3882     STATUS      current
3883     DESCRIPTION
3884         "Additional information about the job's current state, i.e.,
3885         information that augments the value of the job's jmJobState
3886         object.
3887
3888         Implementation of any reason values is OPTIONAL, but an agent
3889         SHOULD return any reason information available. These values
3890         MAY be used with any job state or states for which the reason
3891         makes sense. Since the Job State Reasons will be more dynamic
3892         than the Job State, it is recommended that a job monitoring
3893         application read this object every time jmJobState is read.
3894         When the agent cannot provide a reason for the current state of
3895         the job, the value of the jmJobStateReasons1 object and
3896         jobStateReasonsN attributes SHALL be 0.
3897
3898         The jobStateReasonsN (N=2..4) attributes provide further
3899         additional information about the job's current state."
3900     DEFVAL      { 0 }          -- no reasons
3901     ::= { jmJobEntry 3 }
3902
3903
3904
3905 jmNumberOfInterveningJobs OBJECT-TYPE
3906     SYNTAX      Integer32 (-2..2147483647)
3907     MAX-ACCESS  read-only
3908     STATUS      current
3909     DESCRIPTION
3910         "The number of jobs that are expected to complete processing
3911         before this job has completed processing according to the
3912         implementation's queuing algorithm, if no other jobs were to be
3913         submitted. In other words, this value is the job's queue
3914         position. The agent SHALL return a value of 0 for this
3915         attribute when the job is the next job to complete processing
3916         (or has completed processing)."
```

```
3917     DEFVAL      { 0 }          -- default is no intervening jobs.
3918     ::= { jmJobEntry 4 }
3919
```

```
3920 jmJobKOctetsPerCopyRequested OBJECT-TYPE
3921     SYNTAX      Integer32 (-2..2147483647)
3922     MAX-ACCESS  read-only
3923     STATUS      current
3924     DESCRIPTION
3925         "The total size in K (1024) octets of the document(s) being
3926         requested to be processed in the job.  The agent SHALL round
3927         the actual number of octets up to the next highest K.  Thus 0
3928         octets is represented as '0', 1-1024 octets is represented as
3929         '1', 1025-2048 is represented as '2', etc.
3930
3931         In computing this value, the server/device SHALL NOT include
3932         the multiplicative factors contributed by (1) the number of
3933         document copies, and (2) the number of job copies, independent
3934         of whether the device can process multiple copies of the job or
3935         document without making multiple passes over the job or
3936         document data and independent of whether the output is collated
3937         or not.  Thus the server/device computation is independent of
3938         the implementation and indicates the size of the document(s)
3939         measured in K octets independent of the number of copies."
3940     DEFVAL      { -2 }      -- the default is unknown(-2)
3941     ::= { jmJobEntry 5 }
```

```
3942
3943
3944
3945 jmJobKOctetsProcessed OBJECT-TYPE
3946     SYNTAX      Integer32 (-2..2147483647)
3947     MAX-ACCESS  read-only
3948     STATUS      current
3949     DESCRIPTION
3950         "The total number of octets processed by the server or device
3951         measured in units of K (1024) octets so far.  The agent SHALL
3952         round the actual number of octets processed up to the next
3953         higher K.  Thus 0 octets is represented as '0', 1-1024 octets
3954         is represented as '1', 1025-2048 octets is '2', etc.  For
3955         printing devices, this value is the number interpreted by the
3956         page description language interpreter rather than what has been
3957         marked on media.
3958
3959         For implementations where multiple copies are produced by the
3960         interpreter with only a single pass over the data, the final
3961         value SHALL be equal to the value of the
3962         jmJobKOctetsPerCopyRequested object.  For implementations where
3963         multiple copies are produced by the interpreter by processing
3964         the data for each copy, the final value SHALL be a multiple of
3965         the value of the jmJobKOctetsPerCopyRequested object.
3966
3967         NOTE - See the impressionsCompletedCurrentCopy and
3968         pagesCompletedCurrentCopy attributes for attributes that are
3969         reset on each document copy.
```

```
3970
```

3971 NOTE - The jmJobKOctetsProcessed object can be used with the  
3972 jmJobKOctetsPerCopyRequested object to provide an indication of  
3973 the relative progress of the job, provided that the  
3974 multiplicative factor is taken into account for some  
3975 implementations of multiple copies."  
3976 DEFVAL { 0 } -- default is no octets processed.  
3977 ::= { jmJobEntry 6 }  
3978  
3979

3980 jmJobImpressionsPerCopyRequested OBJECT-TYPE  
3981 SYNTAX Integer32 (-2..2147483647)  
3982 MAX-ACCESS read-only  
3983 STATUS current  
3984 DESCRIPTION  
3985 "The total size in number of impressions of the document(s)  
3986 submitted.  
3987  
3988 In computing this value, the server/device SHALL NOT include  
3989 the multiplicative factors contributed by (1) the number of  
3990 document copies, and (2) the number of job copies, independent  
3991 of whether the device can process multiple copies of the job or  
3992 document without making multiple passes over the job or  
3993 document data and independent of whether the output is collated  
3994 or not. Thus the server/device computation is independent of  
3995 the implementation and reflects the size of the document(s)  
3996 measured in impressions independent of the number of copies.  
3997  
3998 See the definition of the term 'impression' in Section 2."  
3999 DEFVAL { -2 } -- default is unknown(-2)  
4000 ::= { jmJobEntry 7 }  
4001  
4002

4003 jmJobImpressionsCompleted OBJECT-TYPE  
4004 SYNTAX Integer32 (-2..2147483647)  
4005 MAX-ACCESS read-only  
4006 STATUS current  
4007 DESCRIPTION  
4008 "The total number of impressions completed for this job so far.  
4009 For printing devices, the impressions completed includes  
4010 interpreting, marking, and stacking the output. For other  
4011 types of job services, the number of impressions completed  
4012 includes the number of impressions processed.  
4013  
4014 NOTE - See the impressionsCompletedCurrentCopy and  
4015 pagesCompletedCurrentCopy attributes for attributes that are  
4016 reset on each document copy.  
4017  
4018 NOTE - The jmJobImpressionsCompleted object can be used with  
4019 the jmJobImpressionsPerCopyRequested object to provide an  
4020 indication of the relative progress of the job, provided that  
4021 the multiplicative factor is taken into account for some  
4022 implementations of multiple copies.



```
4023
4024     See the definition of the term 'impression' in Section 2 and
4025     the counting example in Section 3.4 entitled 'Monitoring Job
4026     Progress'."
4027     DEFVAL      { 0 }      -- default is no octets
4028     ::= { jmJobEntry 8 }
4029
4030
4031
4032 jmJobOwner OBJECT-TYPE
4033     SYNTAX      JmJobStringTC (SIZE(0..63))
4034     MAX-ACCESS  read-only
4035     STATUS      current
4036     DESCRIPTION
4037         "The coded character set name of the user that submitted the
4038         job.  The method of assigning this user name will be system
4039         and/or site specific but the method MUST ensure that the name
4040         is unique to the network that is visible to the client and
4041         target device.
4042
4043         This value SHOULD be the most authenticated name of the user
4044         submitting the job.
4045
4046         See the OBJECT compliance macro for the minimum maximum length
4047         required for conformance."
4048     DEFVAL      { ''H }      -- default is empty string
4049     ::= { jmJobEntry 9 }
4050
4051
4052
4053
```

```
4054 -- The Attribute Group (MANDATORY)
4055
4056 -- The jmAttributeGroup consists entirely of the jmAttributeTable.
4057 --
4058 -- Implementation of the objects in this group is MANDATORY.
4059 -- See Section 3.1 entitled 'Conformance Considerations'.
4060 -- An agent SHALL implement any attribute if (1) the server or device
4061 -- supports the functionality represented by the attribute and (2) the
4062 -- information is available to the agent.
4063
4064 jmAttribute OBJECT IDENTIFIER ::= { jobmonMIBObjects 4 }
4065
4066
4067
4068 jmAttributeTable OBJECT-TYPE
4069     SYNTAX          SEQUENCE OF JmAttributeEntry
4070     MAX-ACCESS     not-accessible
4071     STATUS          current
4072     DESCRIPTION
4073         "The jmAttributeTable SHALL contain attributes of the job and
4074         document(s) for each job in a job set.  Instead of allocating
4075         distinct objects for each attribute, each attribute is
4076         represented as a separate row in the jmAttributeTable.
4077
4078         The MANDATORY-GROUP macro specifies that this group is
4079         MANDATORY.  An agent SHALL implement any attribute if (1) the
4080         server or device supports the functionality represented by the
4081         attribute and (2) the information is available to the agent. "
4082     ::= { jmAttribute 1 }
4083
4084
4085
4086 jmAttributeEntry OBJECT-TYPE
4087     SYNTAX          JmAttributeEntry
4088     MAX-ACCESS     not-accessible
4089     STATUS          current
4090     DESCRIPTION
4091         "Attributes representing information about the job and
4092         document(s) or resources required and/or consumed.
4093
4094         Each entry in the jmAttributeTable is a per-job entry with an
4095         extra index for each type of attribute (jmAttributeTypeIndex)
4096         that a job can have and an additional index
4097         (jmAttributeInstanceIndex) for those attributes that can have
4098         multiple instances per job.  The jmAttributeTypeIndex object
4099         SHALL contain an enum type that indicates the type of attribute
4100         (see the JmAttributeTypeTC textual-convention).  The value of
4101         the attribute SHALL be represented in either the
4102         jmAttributeValueAsInteger or jmAttributeValueAsOctets objects,
4103         and/or both, as specified in the JmAttributeTypeTC textual-
4104         convention.
4105
```

4106           The agent SHALL create rows in the jmAttributeTable as the  
4107           server or device is able to discover the attributes either from  
4108           the job submission protocol itself or from the document PDL.  
4109           As the documents are interpreted, the interpreter MAY discover  
4110           additional attributes and so the agent adds additional rows to  
4111           this table. As the attributes that represent resources are  
4112           actually consumed, the usage counter contained in the  
4113           jmAttributeValueAsInteger object is incremented according to  
4114           the units indicated in the description of the JmAttributeTypeTC  
4115           enum.

4116

4117           The agent SHALL maintain each row in the jmAttributeTable for  
4118           at least the minimum time after a job completes as specified by  
4119           the jmGeneralAttributePersistence object.

4120

4121           Zero or more entries SHALL exist in this table for each job in  
4122           a job set.

4123

4124           See Section 3.3 entitled 'The Attribute Mechanism' for a  
4125           description of the jmAttributeTable."

4126           INDEX { jmGeneralJobSetIndex, jmJobIndex, jmAttributeTypeIndex,  
4127           jmAttributeInstanceIndex }  
4128           ::= { jmAttributeTable 1 }

4129

4130   JmAttributeEntry ::= SEQUENCE {  
4131       jmAttributeTypeIndex                    JmAttributeTypeTC,  
4132       jmAttributeInstanceIndex               Integer32 (1..32767),  
4133       jmAttributeValueAsInteger               Integer32 (-2..2147483647),  
4134       jmAttributeValueAsOctets                OCTET STRING(SIZE(0..63))  
4135   }  
4136

```
4137 jmAttributeTypeIndex OBJECT-TYPE
4138     SYNTAX      JmAttributeTypeTC
4139     MAX-ACCESS  not-accessible
4140     STATUS      current
4141     DESCRIPTION
4142         "The type of attribute that this row entry represents.
4143
4144         The type MAY identify information about the job or document(s)
4145         or MAY identify a resource required to process the job before
4146         the job start processing and/or consumed by the job as the job
4147         is processed.
4148
4149         Examples of job attributes (i.e., apply to the job as a whole)
4150         that have only one instance per job include:
4151         jobCopiesRequested(90), documentCopiesRequested(92),
4152         jobCopiesCompleted(91), documentCopiesCompleted(93), while
4153         examples of job attributes that may have more than one instance
4154         per job include: documentFormatIndex(37), and
4155         documentFormat(38).
4156
4157         Examples of document attributes (one instance per document)
4158         include: fileName(34), and documentName(35).
4159
4160         Examples of required and consumed resource attributes include:
4161         pagesRequested(130), mediumRequested(170), pagesCompleted(131),
4162         and mediumConsumed(171), respectively."
4163 ::= { jmAttributeEntry 1 }
4164
4165
4166
4167 jmAttributeInstanceIndex OBJECT-TYPE
4168     SYNTAX      Integer32 (1..32767)
4169     MAX-ACCESS  not-accessible
4170     STATUS      current
4171     DESCRIPTION
4172         "A running 16-bit index of the attributes of the same type for
4173         each job.  For those attributes with only a single instance per
4174         job, this index value SHALL be 1.  For those attributes that
4175         are a single value per document, the index value SHALL be the
4176         document number, starting with 1 for the first document in the
4177         job.  Jobs with only a single document SHALL use the index
4178         value of 1.  For those attributes that can have multiple values
4179         per job or per document, such as documentFormatIndex(37) or
4180         documentFormat(38), the index SHALL be a running index for the
4181         job as a whole, starting at 1."
4182 ::= { jmAttributeEntry 2 }
4183
```

```
4184 jmAttributeValueAsInteger OBJECT-TYPE
4185     SYNTAX      Integer32 (-2..2147483647)
4186     MAX-ACCESS  read-only
4187     STATUS      current
4188     DESCRIPTION
4189         "The integer value of the attribute.  The value of the
4190         attribute SHALL be represented as an integer if the enum
4191         description in the JmAttributeTypeTC textual-convention
4192         definition has the tag: 'INTEGER:'.
```

4193

4194 Depending on the enum definition, this object value MAY be an  
4195 integer, a counter, an index, or an enum, depending on the  
4196 jmAttributeTypeIndex value. The units of this value are  
4197 specified in the enum description.

4198

4199 For those attributes that are accumulating job consumption as  
4200 the job is processed as specified in the JmAttributeTypeTC  
4201 textual-convention, SHALL contain the final value after the job  
4202 completes processing, i.e., this value SHALL indicate the total  
4203 usage of this resource made by the job.

4204

4205 A monitoring application is able to copy this value to a  
4206 suitable longer term storage for later processing as part of an  
4207 accounting system.

4208

4209 Since the agent MAY add attributes representing resources to  
4210 this table while the job is waiting to be processed or being  
4211 processed, which can be a long time before any of the resources  
4212 are actually used, the agent SHALL set the value of the  
4213 jmAttributeValueAsInteger object to 0 for resources that the  
4214 job has not yet consumed.

4215

4216 Attributes for which the concept of an integer value is  
4217 meaningless, such as fileName(34), jobName, and  
4218 processingMessage, do not have the 'INTEGER:' tag in the  
4219 JmAttributeTypeTC definition and so an agent SHALL always  
4220 return a value of '-1' to indicate 'other' for the value of the  
4221 jmAttributeValueAsInteger object for these attributes.

4222

4223 For attributes which do have the 'INTEGER:' tag in the  
4224 JmAttributeTypeTC definition, if the integer value is not (yet)  
4225 known, the agent either (1) SHALL not materialize the row in  
4226 the jmAttributeTable until the value is known or (2) SHALL  
4227 return a '-2' to represent an 'unknown' counting integer value,  
4228 a '0' to represent an 'unknown' index value, and a '2' to  
4229 represent an 'unknown(2)' enum value."

```
4230     DEFVAL      { -2 }      -- default value is unknown(-2)
4231     ::= { jmAttributeEntry 3 }
```

4232

```
4233 jmAttributeValueAsOctets OBJECT-TYPE
4234     SYNTAX      OCTET STRING(SIZE(0..63))
4235     MAX-ACCESS  read-only
4236     STATUS      current
4237     DESCRIPTION
4238         "The octet string value of the attribute.  The value of the
4239         attribute SHALL be represented as an OCTET STRING if the enum
4240         description in the JmAttributeTypeTC textual-convention
4241         definition has the tag: 'OCTETS:'.
4242
4243         Depending on the enum definition, this object value MAY be a
4244         coded character set string (text), such as 'JmUTF8StringTC', or
4245         a binary octet string, such as 'DateAndTime'.
4246
4247         Attributes for which the concept of an octet string value is
4248         meaningless, such as pagesCompleted, do not have the tag
4249         'OCTETS:' in the JmAttributeTypeTC definition and so the agent
4250         SHALL always return a zero length string for the value of the
4251         jmAttributeValueAsOctets object.
4252
4253         For attributes which do have the 'OCTETS:' tag in the
4254         JmAttributeTypeTC definition, if the OCTET STRING value is not
4255         (yet) known, the agent either SHALL NOT materialize the row in
4256         the jmAttributeTable until the value is known or SHALL return a
4257         zero-length string."
4258     DEFVAL      { ''H }      -- empty string
4259     ::= { jmAttributeEntry 4 }
4260
```

```
4261 -- Notifications and Trapping
4262 -- Reserved for the future
4263
4264 jobmonMIBNotifications OBJECT IDENTIFIER ::= { jobmonMIB 2 }
4265
4266
4267
4268 -- Conformance Information
4269
4270 jmMIBConformance OBJECT IDENTIFIER ::= { jobmonMIB 3 }
4271
4272
4273
4274 -- compliance statements
4275 jmMIBCompliance MODULE-COMPLIANCE
4276     STATUS current
4277     DESCRIPTION
4278         "The compliance statement for agents that implement the
4279         job monitoring MIB."
4280     MODULE -- this module
4281     MANDATORY-GROUPS {
4282         jmGeneralGroup, jmJobIDGroup, jmJobGroup, jmAttributeGroup }
4283
4284     OBJECT jmGeneralJobSetName
4285     SYNTAX JmUTF8StringTC (SIZE(0..8))
4286     DESCRIPTION
4287         "Only 8 octets maximum string length NEED be supported by the
4288         agent."
4289
4290     OBJECT jmJobOwner
4291     SYNTAX JmJobStringTC (SIZE(0..16))
4292     DESCRIPTION
4293         "Only 16 octets maximum string length NEED be supported by the
4294         agent."
4295
4296 -- There are no CONDITIONALLY MANDATORY or OPTIONAL groups.
4297
4298 ::= { jmMIBConformance 1 }
4299
```

```
4300 jmMIBGroups      OBJECT IDENTIFIER ::= { jmMIBConformance 2 }
4301
4302 jmGeneralGroup OBJECT-GROUP
4303     OBJECTS {
4304         jmGeneralNumberOfActiveJobs,    jmGeneralOldestActiveJobIndex,
4305         jmGeneralNewestActiveJobIndex,  jmGeneralJobPersistence,
4306         jmGeneralAttributePersistence,  jmGeneralJobSetName}
4307     STATUS current
4308     DESCRIPTION
4309         "The general group."
4310     ::= { jmMIBGroups 1 }
4311
4312
4313
4314 jmJobIDGroup OBJECT-GROUP
4315     OBJECTS {
4316         jmJobIDJobSetIndex, jmJobIDJobIndex }
4317     STATUS current
4318     DESCRIPTION
4319         "The job ID group."
4320     ::= { jmMIBGroups 2 }
4321
4322
4323
4324 jmJobGroup OBJECT-GROUP
4325     OBJECTS {
4326         jmJobState, jmJobStateReasons1, jmNumberOfInterveningJobs,
4327         jmJobKOctetsPerCopyRequested, jmJobKOctetsProcessed,
4328         jmJobImpressionsPerCopyRequested, jmJobImpressionsCompleted,
4329         jmJobOwner }
4330     STATUS current
4331     DESCRIPTION
4332         "The job group."
4333     ::= { jmMIBGroups 3 }
4334
4335
4336
4337 jmAttributeGroup OBJECT-GROUP
4338     OBJECTS {
4339         jmAttributeValueAsInteger, jmAttributeValueAsOctets }
4340     STATUS current
4341     DESCRIPTION
4342         "The attribute group."
4343     ::= { jmMIBGroups 4 }
4344
4345
4346 END
```



## 4347 5 Appendix A - Implementing the Job Life Cycle

4348 The job object has well-defined states and client operations that  
4349 affect the transition between the job states. Internal server and  
4350 device actions also affect the transitions of the job between the job  
4351 states. These states and transitions are referred to as the job's *life*  
4352 *cycle*.

4353 Not all implementations of job submission protocols have all of the  
4354 states of the job model specified here. The job model specified here  
4355 is intended to be a superset of most implementations. It is the  
4356 purpose of the agent to map the particular implementation's job life  
4357 cycle onto the one specified here. The agent MAY omit any states not  
4358 implemented. Only the processing and completed states are required to  
4359 be implemented by an agent. However, a conforming management  
4360 application SHALL be prepared to accept any of the states in the job  
4361 life cycle specified here, so that the management application can  
4362 interoperate with any conforming agent.

4363 The job states are intended to be user visible. The agent SHALL make  
4364 these states visible in the MIB, but only for the subset of job states  
4365 that the implementation has. Some implementations MAY need to have  
4366 sub-states of these user-visible states. The jmJobStateReasons1 object  
4367 and the jobStateReasonsN ( $N=2..4$ ) attributes can be used to represent  
4368 the sub-states of the jobs.

4369 Job states are intended to last a user-visible length of time in most  
4370 implementations. However, some jobs may pass through some states in  
4371 zero time in some situations and/or in some implementations.

4372 The job model does not specify how accounting and auditing is  
4373 implemented, except to assume that accounting and auditing logs are  
4374 separate from the job life cycle and last longer than job entries in  
4375 the MIB. Jobs in the completed, aborted, or canceled states are not  
4376 logs, since jobs in these states are accessible via SNMP protocol  
4377 operations and SHALL be removed from the Job Monitoring MIB tables  
4378 after a site-settable or implementation-defined period of time. An  
4379 accounting application MAY copy accounting information incrementally to  
4380 an accounting log as a job processes, or MAY be copied while the job is  
4381 in the canceled, aborted, or completed states, depending on  
4382 implementation. The same is true for auditing logs.

4383 The jmJobState object specifies the standard job states. The normal  
4384 job state transitions are shown in the state transition diagram  
4385 presented in Table 1.

## 4386 6 APPENDIX B - Support of Job Submission Protocols

4387 A companion PWG document, entitled "Job Submission Protocol Mapping  
4388 Recommendations for the Job Monitoring MIB" [protomap] contains the  
4389 recommended usage of each of the objects and attributes in this MIB  
4390 with a number of job submission protocols. In particular, which job  
4391 submission ID format should be used is indicated for each job  
4392 submission protocol.

4393 Some job submission protocols have support for the client to specify a  
4394 job submission ID. A second approach is to enhance the document format  
4395 to embed the job submission ID in the document data. This second  
4396 approach is independent of the job submission protocol. This appendix  
4397 lists some examples of these approaches.

4398 Some PJJ implementations wrap a banner page as a PJJ job around a job  
4399 submitted by a client. If this results in multiple job submission IDs,  
4400 the agent SHALL create multiple jmJobIDEntry rows in the jmJobIDTable  
4401 that each point to the same job entry in the job tables. See the  
4402 specification of the jmJobIDEntry.

## 4403 7 References

4404 [char-set-policy] Harald Avelstrand, "IETF Policy on Character Sets and  
4405 Language", June 1997. Latest draft: <draft-avelstrand-charset-  
4406 policy-00.txt>

4407 [GB2312] GB 2312-1980, "Chinese People's Republic of China (PRC) mixed  
4408 one byte and two byte coded character set"

4409 [hr-mib] P. Grillo, S. Waldbusser, "Host Resources MIB", RFC 1514,  
4410 September 1993

4411 [iana] J. Reynolds, and J. Postel, "Assigned Numbers", STD 2, RFC 1700,  
4412 ISI, October 1994.

4413 [IANA-charsets] Coded Character Sets registered by IANA and assigned an  
4414 enum value for use in the CodedCharSet textual convention imported from  
4415 the Printer MIB. See ftp://ftp.isi.edu/in-  
4416 notes/iana/assignments/character-sets

4417 [iana-media-types] IANA Registration of MIME media types (MIME content  
4418 types/subtypes). See ftp://ftp.isi.edu/in-notes/iana/assignments/

4419 [ipp-model] Internet Printing Protocol/1.0: Model and Semantics, work  
4420 in progress on the IETF standards track. See draft-ietf-ipp-model-  
4421 09.txt. See also http://www.pwg.org/ipp/index.html

4422 [ISO-639] ISO 639:1988 (E/F) - Code for Representation of names of  
4423 languages - The International Organization for Standardization, 1st  
4424 edition, 1988.

- 4425 [ISO 646] ISO/IEC 646:1991, "Information technology -- ISO 7-bit coded  
4426 character set for information interchange", JTC1/SC2.
- 4427 [ISO 8859] ISO/IEC 8859-1:1987, "Information technology -- 8-bit single  
4428 byte coded graphic character sets - Part 1: Latin alphabet No. 1,  
4429 JTC1/SC2."
- 4430 [ISO 2022] ISO/IEC 2022:1994 - "Information technology -- Character  
4431 code structure and extension techniques", JTC1/SC2.
- 4432 [ISO-3166] ISO 3166:1988 (E/F) - Codes for representation of names of  
4433 countries - The International Organization for Standardization, 3rd  
4434 edition, 1988-08-15."
- 4435 [ISO-10646] ISO/IEC 10646-1:1993, "Information technology -- Universal  
4436 Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and  
4437 Basic Multilingual Plane, JTC1/SC2.
- 4438 [iso-dpa] ISO/IEC 10175 Document Printing Application (DPA). See  
4439 <ftp://ftp.pwg.org/pub/pwg/dpa/>
- 4440 [JIS X0208] JIS X0208-1990, "Japanese two byte coded character set."
- 4441 [mib-II] MIB-II, RFC 1213.
- 4442 [print-mib] Smith, R., Wright, F., Hastings, T., Zilles, S. and  
4443 Gyllenskog, J., "Printer MIB", RFC 1759, proposed IETF standard, March  
4444 1995. See also [print-mib-draft].
- 4445 [print-mib-draft] Turner, R., "Printer MIB", work in progress, on the  
4446 standards track as a draft standard: <draft-ietf-printmib-mib-info-  
4447 02.txt>, October 15, 1997.
- 4448 [protomap] Bergman, R., "Job Submission Protocol Mapping  
4449 Recommendations for the Job Monitoring MIB," work in progress as an  
4450 informational RFC. See <draft-bergman-printmib-job-protomap-01.txt>,  
4451 January 12, 1998.
- 4452 [pwg] The Printer Working Group is a printer industry consortium open  
4453 to any individuals. For more information, access the PWG web page:  
4454 <http://www.pwg.org>
- 4455 [req-words] S. Bradner, "Keywords for use in RFCs to Indicate  
4456 Requirement Levels", RFC 2119, March 1997.
- 4457 [rfc 1738] Berners-Lee, T., Masinter, L., McCahill, M., "Uniform  
4458 Resource Locators (URL)", RFC 1738, December 1994.
- 4459 [RFC-1766] Avelstrand, H., "Tags for the Identification of Languages",  
4460 RFC 1766, March 1995.

4461 [rfc 2130] C. Weider, C. Preston, K. Simonsen, H. Alvestrand, R.  
4462 Atkinson, M. Crispin, and P. Svanberg, "The Report of the IAB Character  
4463 Set Workshop held 29 Feb-1 March, 1997", April 1997, RFC 2130.

4464 [SMIv2-SMI] J. Case, et al. "Structure of Management Information for  
4465 Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC  
4466 1902, January 1996.

4467 [SMIv2-TC] J. Case, et al. "Textual Conventions for Version 2 of the  
4468 Simple Network Management Protocol (SNMPv2)", RFC 1903, January 1996.

4469 [tipsi] IEEE 1284.1, Transport-independent Printer System Interface  
4470 (TIPSI).

4471 [URI-spec] Berners-Lee, T., Masinter, L., McCahill, M. , "Uniform  
4472 Resource Locators (URL)", RFC 1738, December, 1994.

4473 [US-ASCII] Coded Character Set - 7-bit American Standard Code for  
4474 Information Interchange, ANSI X3.4-1986.

4475 [UTF-8] F. Yergeau, "UTF-8, a transformation format of Unicode and ISO  
4476 10646", RFC 2044, October 1996.

4477 8 Author's Addresses  
4478     Ron Bergman  
4479     Dataproducts Corp.  
4480     1757 Tapo Canyon Road  
4481     Simi Valley, CA 93063-3394  
4482  
4483     Phone: 805-578-4421  
4484     Fax: 805-578-4001  
4485     Email: rbergman@dpc.com  
4486  
4487  
4488     Tom Hastings  
4489     Xerox Corporation, ESAE-231  
4490     701 S. Aviation Blvd.  
4491     El Segundo, CA 90245  
4492  
4493     Phone: 310-333-6413  
4494     Fax: 310-333-5514  
4495     EMail: hastings@cp10.es.xerox.com  
4496  
4497  
4498     Scott A. Isaacson  
4499     Novell, Inc.  
4500     122 E 1700 S  
4501     Provo, UT 84606  
4502  
4503     Phone: 801-861-7366  
4504     Fax: 801-861-4025

4505        EMail: scott\_isaacson@novell.com

4506

4507

4508        Harry Lewis

4509        IBM Corporation

4510        6300 Diagonal Hwy

4511        Boulder, CO 80301

4512

4513        Phone: (303) 924-5337

4514        Fax:

4515        Email: harryl@us.ibm.com

4516

4517

4518        Send questions and comments to the Printer Working Group (PWG)  
4519        using the Job Monitoring Project (JMP) Mailing List: jmp@pwg.org

4520

4521        To learn how to subscribe, send email to: jmp-request@pwg.org

4522

4523        Implementers of this specification are encouraged to join the jmp  
4524        mailing list in order to participate in discussions on any  
4525        clarifications needed and registration proposals for additional  
4526        attributes and values being reviewed in order to achieve consensus.

4527

4528        For further information, access the PWG web page under "JMP":

4529

4530        <http://www.pwg.org/>

4531

4532        Other Participants:

4533        Chuck Adams - Tektronix

4534        Jeff Barnett - IBM

4535        Keith Carter, IBM Corporation

4536        Jeff Copeland - QMS

4537        Andy Davidson - Tektronix

4538        Roger deBry - IBM

4539        Mabry Dozier - QMS

4540        Lee Farrell - Canon

4541        Steve Gebert - IBM

4542        Robert Herriot - Sun Microsystems Inc.

4543        Shige Kanemitsu - Kyocera

4544        David Kellerman - Northlake Software

4545        Rick Landau - Digital

4546        Pete Loya - HP

4547        Ray Lutz - Cognisys

4548        Jay Martin - Underscore

4549        Mike MacKay, Novell, Inc.

4550        Stan McConnell - Xerox

4551        Carl-Uno Manros, Xerox, Corp.

4552        Pat Nogay - IBM

4553        Bob Pentecost - HP

4554        Rob Rhoads - Intel

4555 David Roach - Unisys  
4556 Stuart Rowley - Kyocera  
4557 Hiroyuki Sato - Canon  
4558 Bob Setterbo - Adobe  
4559 Gail Songer, EFI  
4560 Mike Timperman - Lexmark  
4561 Randy Turner - Sharp  
4562 William Wagner - Digital Products  
4563 Jim Walker - Dazel  
4564 Chris Wellens - Interworking Labs  
4565 Rob Whittle - Novell  
4566 Don Wright - Lexmark  
4567 Lloyd Young - Lexmark  
4568 Atsushi Yuki - Kyocera  
4569 Peter Zehler, Xerox, Corp.

## 4570 9 Change History

4571 This section summarizes the changes in each version after version 1.0  
4572 in reverse chronological order.

### 4573 9.1 Changes to produce version 1.1, dated October 1, 1998

4574 The following changes were made to version 1.0, dated February 3, 1998  
4575 to make version 1.1, dated October 1, 1998:

- 4576 1. Clarified sections 3.3.3 and 3.3.7 so that the DEFVAL of 0 for index  
4577 attributes is different from the DEFVAL for  
4578 jmAttributeValueAsInteger which is -2.
- 4579 2. Clarified the relationships of the values of the  
4580 JmJobCollationTypeTC with the IPP "multiple-document-handling"  
4581 attribute.
- 4582 3. Clarified that the values of the mediumRequested(170) and  
4583 mediumConsumed(171) attributes may be any of the IPP 'media' values  
4584 which are media names, media size names, and input tray names.
- 4585 4. Added the two attributes approved by the PWG for registration in  
4586 April 1998: mediumTypeConsumed(174) and mediumSizeConsumed(175).
- 4587 5. Changed "insure" to "ensure".
- 4588 6. Correct an incorrect reference in the jmAttributeEntry DESCRIPTION  
4589 from jmJobTable to jmAttributeTable.

- 4590 9.2 Changes to produce version 1.2, dated October 2, 1998
- 4591 The following changes were made to version 1.1, dated October 1, 1998  
4592 to make version 1.2, dated October 2, 1998:
- 4593 1. Removed all REFERENCE clauses since they referred to sections in the  
4594 specification that were not in the MIB.
  - 4595 2. Moved the definitions of the attributes from the TC to a new section  
4596 3.3.8.
  - 4597 3. Removed the attributes from the Table of Contents
  - 4598 4. Added the data types as ASN.1 comments after each attribute enum.
  - 4599 5. Changed a number of occurrences of "SHALL" to "is" when they were  
4600 just definitions, rather than conformance requirements.
- 4601

## 4602 10 INDEX

4603 This index includes the textual conventions, the objects, and the  
4604 attributes. Textual conventions all start with the prefix: "JM" and  
4605 end with the suffix: "TC". Objects all starts with the prefix: "jm"  
4606 followed by the group name. Attributes are identified with enums, and  
4607 so start with any lower case letter and have no special prefix.

4608  
4609 colorantConsumed, 41  
4610 colorantRequested, 40  
4611 deviceNameRequested, 30  
4612 documentCopiesCompleted, 35  
4613 documentCopiesRequested, 35  
4614 documentFormat, 32  
4615 documentFormatIndex, 31  
4616 documentName, 31  
4617 fileName, 31  
4618 finishing, 34  
4619 fullColorImpressionsCompleted, 37  
4620 highlightColorImpressionsCompleted, 38  
4621 impressionsCompletedCurrentCopy, 37  
4622 impressionsInterpreted, 37  
4623 impressionsSentToDevice, 37  
4624 impressionsSpooled, 37  
4625 jmAttributeInstanceIndex, 100  
4626 jmAttributeTypeIndex, 100  
4627 JmAttributeTypeTC, 70  
4628 jmAttributeValueAsInteger, 101  
4629 jmAttributeValueAsOctets, 102  
4630 JmBooleanTC, 60  
4631 JmFinishingTC, 58  
4632 jmGeneralAttributePersistence, 87  
4633 jmGeneralJobPersistence, 87  
4634 jmGeneralJobSetIndex, 85  
4635 jmGeneralJobSetName, 88  
4636 jmGeneralNewestActiveJobIndex, 86  
4637 jmGeneralNumberOfActiveJobs, 85  
4638 jmGeneralOldestActiveJobIndex, 86  
4639 JmJobCollationTypeTC, 62  
4640 jmJobIDJobIndex, 91  
4641 jmJobIDJobSetIndex, 91  
4642 jmJobImpressionsCompleted, 96  
4643 jmJobImpressionsPerCopyRequested, 96  
4644 jmJobIndex, 93  
4645 jmJobKOctetsPerCopyRequested, 95  
4646 jmJobKOctetsProcessed, 95  
4647 jmJobOwner, 97  
4648 JmJobServiceTypesTC, 74  
4649 JmJobSourcePlatformTypeTC, 57  
4650 jmJobState, 93  
4651 jmJobStateReasons1, 94



4652 JmJobStateReasons1TC, 75  
4653 JmJobStateReasons2TC, 79  
4654 JmJobStateReasons3TC, 83  
4655 JmJobStateReasons4TC, 83  
4656 JmJobStateTC, 67  
4657 JmJobStringTC, 56  
4658 jmJobSubmissionID, 90  
4659 JmJobSubmissionIDTypeTC, 62  
4660 JmMediumTypeTC, 60  
4661 JmNaturalLanguageTagTC, 56  
4662 jmNumberOfInterveningJobs, 94  
4663 JmPrinterResolutionTC, 59  
4664 JmPrintQualityTC, 59  
4665 JmTimeStampTC, 56  
4666 JmTonerEconomyTC, 60  
4667 JmUTF8StringTC, 56  
4668 jobAccountName, 27  
4669 jobCodedCharSet, 26  
4670 jobCollationType, 36  
4671 jobComment, 31  
4672 jobCompletionTime, 43  
4673 jobCopiesCompleted, 35  
4674 jobCopiesRequested, 35  
4675 jobHold, 33  
4676 jobHoldUntil, 33  
4677 jobKOctetsTransferred, 36  
4678 jobName, 28  
4679 jobNaturalLanguageTag, 27  
4680 jobOriginatingHost, 30  
4681 jobPriority, 32  
4682 jobProcessAfterDateAndTime, 33  
4683 jobProcessingCPUtime, 43  
4684 jobServiceTypes, 29  
4685 jobSourceChannelIndex, 29  
4686 jobSourcePlatformType, 29  
4687 jobStartedBeingHeldTime, 42  
4688 jobStartedProcessingTime, 43  
4689 jobStateReasons2, 25  
4690 jobStateReasons3, 25  
4691 jobStateReasons4, 25  
4692 jobSubmissionTime, 42  
4693 jobSubmissionToServerTime, 42  
4694 jobURI, 27  
4695 mediumConsumed, 40  
4696 mediumRequested, 40  
4697 mediumSizeConsumed, 41  
4698 mediumTypeConsumed, 41  
4699 numberOfDocuments, 30  
4700 other, 24  
4701 outputBin, 33  
4702 pagesCompleted, 38  
4703 pagesCompletedCurrentCopy, 39

4704 pagesRequested, 38  
4705 physicalDevice, 30  
4706 printerResolutionRequested, 34  
4707 printerResolutionUsed, 34  
4708 printQualityRequested, 34  
4709 printQualityUsed, 34  
4710 processingMessage, 25  
4711 processingMessageNaturalLangTag, 26  
4712 queueNameRequested, 30  
4713 serverAssignedJobName, 28  
4714 sheetCompletedCopyNumber, 36  
4715 sheetCompletedDocumentNumber, 36  
4716 sheetsCompleted, 39  
4717 sheetsCompletedCurrentCopy, 39  
4718 sheetsRequested, 39  
4719 sides, 34  
4720 submittingApplicationName, 29  
4721 submittingServerName, 29  
4722 tonerDensityRequested, 34  
4723 tonerDensityUsed, 35  
4724 tonerEcomonyRequested, 34  
4725 tonerEcomonyUsed, 34  
  
4726