1	Job Monitoring MIB, V0.85
2	(This cover page is <i>not</i> part of the Internet-Draft)
3	
4	From: Tom Hastings
5	Date: 08/08/97
6	Version: 0.85
7	File: ftp://ftp.pwg.org/pub/jmp/mibs/jmp-mib.doc .pdf jmp-mibr.doc .pdf .pdr
8 9	Status: Eighth draft MIB that incorporates the resolutions of issues 110 to 120 from the 8/8 JMP meeting. See the change history in the separate file: changes.doc .pdf.
10 11	We agreed that the MIB specification is finished except for any editorial comments that people may have. See the separate issues.doc and .pdf file.
12 13 14	I've also produced a variation on this document which has all variable font (jmp-mib.doc .pdf) without revision marks. This is the version that the JMP should use to make comments. It has line numbers.
15	The MIB has been greatly simplified so that now there are only 18 objects in the MIB.

16 There are 65 attributes.

INTERNET-DRAFT Ron Bergman Dataproducts Corp. Tom Hastings Xerox Corporation Scott Isaacson Novell, Inc. Harry Lewis
IBM Corp. August 8, 1997
Job Monitoring MIB - V0.85
<draft-ietf-printmib-job-monitor-05.txt></draft-ietf-printmib-job-monitor-05.txt>
Expires Feb 8, 1997
Status of this Memo
This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.
Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."
To learn the current status of any Internet-Draft, please check the "1id- abstracts.txt" listing contained in the Internet-Drafts Shadow Directories on ftp.is.co.za (Africa), nic.nordu.net (Europe), munnari.oz.au (Pacific Rim), ds.internic.net (US East Coast), or ftp.isi.edu (US West Coast).
Abstract
This Internet-Draft specifies a small set of read-only SNMP MIB objects for (1) monitoring the status and progress of print jobs (2) obtaining resource requirements before a job is processed, (3) monitoring resource consumption while a job is being processed and (4) collecting resource accounting data after the completion of a job. This MIB is intended to be implemented (1) in a printer or (2) in a server that supports one or more printers. Use of the object set is not limited to printing. However, support for services other than printing is outside the scope of this Job Monitoring MIB. Future extensions to this MIB may include, but are not limited to, fax machines and scanners.

Job	Monitoring	MIB,	V0.85
-----	------------	------	-------

53		
54	TABLE OF CONTENTS	
54		
55	1. INTRODUCTION	9
56	1.1 Types of Information in the MIB	9
57	1.2 Types of Job Monitoring Applications	10
58	2. TERMINOLOGY AND JOB MODEL	11
59	2.1 System Configurations for the Job Monitoring MIB	14
60	2.1.1 Configuration 1 - client-printer	14
61	2.1.2 Configuration 2 - client-server-printer - agent in the server	15
62	2.1.3 Configuration 3 - client-server-printer - client monitors printer agent and server	16
63	3. MANAGED OBJECT USAGE	18
64	3.1 Conformance Considerations	18
65	3.1.1 Conformance Terminology	
66	3.1.2 Agent Conformance Requirements	18
67	3.1.2.1 MIB II System Group objects	19
68	3.1.2.2 MIB II Interface Group objects	19
69	3.1.2.3 Printer MIB objects	19
70	3.1.3 Job Monitoring Application Conformance Requirements	19
71	3.2 The Job Tables and the Oldest Active and Newest Active Indexes	20
72	3.3 The Attribute Mechanism	
73	3.3.1 Conformance of Attribute Implementation	
74	3.3.2 Useful, 'Unknown', and 'Other' Values for Objects and Attributes	
75	3.3.3 Data Sub-types and Attribute Naming Conventions	
76	3.3.4 Single-Value (Row) Versus Multi-Value (MULTI-ROW) Attributes	
77	3.3.5 Requested Attributes	
78	3.3.6 Consumption Attributes	
79	3.3.7 Index Value Attributes	25
80	3.4 Job Identification	25
81	3.5 Internationalization Considerations	
82	3.5.1 'JmUTF8StringTC' for text generated by the server or device	
83	3.5.2 'JmJobStringTC' for text generated by the job submitter	
84	3.5.3 'DateAndTime' for representing the date and time	25
85	3.6 IANA Considerations	
86	3.6.1 IANA Registration of enums	27

87	3.6.1.1 Type 1 enumerations	
88	3.6.1.2 Type 2 enumerations	
89	3.6.1.3 Type 3 enumeration	
90	3.6.2 IANA Registration of type 2 bit values	
91	3.6.3 IANA Registration of Job Submission Id Formats	
92	3.6.4 IANA Registration of MIME types/sub-types for document-formats	
93	3.7 Security Considerations	29
94	3.7.1 Read-Write objects	
95	3.7.2 Read-Only Objects In Other User's Jobs	29
96	3.8 Values for Objects	23
97	3.9 Notifications	30
98	4. MIB SPECIFICATION	
99	Textual conventions for this MIB module	
100	JmUTF8StringTC	
101	JmJobStringTC	
102	JmTimeStampTC	
103	JmJobSourcePlatformTypeTC	
104	JmFinishingTC	
105	JmPrintQualityTC	
106	JmPrinterResolutionTC	
107	JmTonerEconomyTC	
108	JmBooleanTC	
109	JmMediumTypeTC	
110	JmJobSubmissionIDTypeTC	
111	JmJobStateTC	
112	JmAttributeTypeTC	
113	other (Int32(-2) and/or Octets63)	
114	Job State attributes	
115	jobStateReasons2 (JmJobStateReasons2TC)	
116	jobStateReasons3 (JmJobStateReasons3TC)	
117	jobStateReasons4 (JmJobStateReasons4TC)	
118	processingMessage (UTF8String63)	
119	jobCodedCharSet (CodedCharSet)	
120 121	Job Identification attributes	
121	jobAccountName (JobString63)	
122	serverAssignedJobName (JobString63)	
123	jobName (JobString63) jobServiceTypes (JmJobServiceTypesTC)	
124	jobSourceChannelIndex (Int32(0))	
125	jobSourcePlatformType (JmJobSourcePlatformTypeTC)	
120	submittingServerName (JobString63)	
127	submittingApplicationName (JobString63)	
120	jobOriginatingHost (JobString63)	
	J 	

	30	deviceNameRequested (JobString63)	46
	31	queueNameRequested (JobString63)	
	32	physicalDevice (hrDeviceIndex and/or UTF8String63)	46
	33	numberOfDocuments (Int32(-2))	47
1	34	fileName (JobString63)	47
1	35	documentName (JobString63)	47
1	36	jobComment (JobString63)	47
1	37	documentFormatIndex (Int32(0))	47
1	38	documentFormat (PrtInterpreterLangFamilyTC and/or Octets63)	47
1	39	Job Parameter attributes	
1	40	jobPriority (Int32(1.100))	
1	41	jobProcessAfterDateAndTime (DateAndTime)	
	42	jobHold (JmBooleanTC)	
	43	jobHoldUntil (JobString63)	
	44	outputBin (Int32(0) and/or JobString63)	
	45	sides (Int32(-22))	
	46	finishing (JmFinishingTC)	
	47	Image Quality attributes (requested and used)	
	48	printQualityRequested (JmPrintQualityTC)	
	49	printQualityUsed (JmPrintQualityTC)	
	50	printerResolutionRequested (JmPrinterResolutionTC)	
	51	printerResolutionUsed (JmPrinterResolutionTC)	
	52	tonerEcomonyRequested (JmTonerEconomyTC)	
	53	tonerEcomonyUsed (JmTonerEconomyTC)	
	54	tonerDensityRequested (Int32(-2100))	
	55	tonerDensityUsed (Int32(-2100))	
	56	Job Progress attributes (requested and consumed)	
	57	jobCopiesRequested (Int32(-2))	
	58	jobCopiesCompleted (Int32(-2))	
	59	documentCopiesRequested (Int32(-2))	
	60	documentCopiesCompleted (Int32(-2))	
	61	jobKOctetsTransferred (Int32(-2)).	
	62	Impression attributes (requested and consumed)	
	63	impressionsSpooled (Int32(-2))	
	64	impressionsSentToDevice (Int32(-2))	
	65	impressionsInterpreted (Int32(-2))	
	66	impressionsCompletedCurrentCopy (Int32(-2))	
	67	fullColorImpressionsCompleted (Int32(-2))	
	68	highlightColorImpressionsCompleted (Int32(-2))	
	69	Page attributes (requested and consumed)	
	70	pagesRequested (Int32(-2))	
	71	pagesCompleted (Int32(-2))	
	72	pagesCompletedCurrentCopy (Int32(-2))	
	73	Sheet attributes (requested and consumed)	
	74	sheetsRequested (Int32(-2))	
	75	sheetsCompleted (Int32(-2))	
	76	sheetsCompletedCurrentCopy (Int32(-2))	
	77	Resource attributes (requested and consumed)	
	78	mediumRequested (JmMediumTypeTC and/or JobString63)	
-			

179	mediumConsumed (JobString63)	
180	colorantRequested (Int32(-2) and/or JobString63)	53
181	colorantConsumed (Int32(-2) and/or JobString63)	54
182	Time attributes (set by server or device)	54
183	jobSubmissionToServerTime (JmTimeStampTC and/or DateAndTime)	54
184	jobSubmissionTime (JmTimeStampTC and/or DateAndTime)	
185	jobStartedBeingHeldTime (JmTimeStampTC and/or DateAndTime)	55
186	jobStartedProcessingTime (JmTimeStampTC and/or DateAndTime)	55
187	jobCompletedTime (JmTimeStampTC and/or DateAndTime)	55
188	jobProcessingCPUTime (Int32(-2))	55
189	JmJobServiceTypesTC	57
190	JmJobStateReasons1TC	
191	JmJobStateReasons2TC	61
192	JmJobStateReasons3TC	64
193	JmJobStateReasons4TC	65
194	The General Group (MANDATORY)	66
195	jmGeneralJobSetIndex (Int32(132767))	66
196	jmGeneralNumberOfActiveJobs (Int32(0))	67
197	jmGeneralOldestActiveJobIndex (Int32(0))	67
198	jmGeneralNewestActiveJobIndex (Int32(0))	67
199	jmGeneralJobPersistence (Int32(15))	68
200	jmGeneralAttributePersistence (Int32(15))	68
201	jmGeneralJobSetName (UTF8String63)	
201	jinooneraijooben taine (011 obumg03)	
202	The Job ID Group (MANDATORY)	
202 203		69
202 203 204	The Job ID Group (MANDATORY)	69
202 203	The Job ID Group (MANDATORY) jmJobSubmissionID (OCTET STRING(SIZE(48)))	69
202 203 204 205 206	The Job ID Group (MANDATORY)	69 70 70 70
202 203 204 205 206 207	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209 210	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209 210 211	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209 210 211 212	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209 210 211 212 213	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209 210 211 212 213 214	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209 210 211 212 213	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209 210 211 212 213 214 215 216	The Job ID Group (MANDATORY)	69 70 70 70 71 72 72 72 72 73 73 73 73 74 74 74 75 75
202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217	The Job ID Group (MANDATORY)	69 70 70 70 70 71 72 72 72 72 72 73 73 73 73 73 74 74 74 75 75 76
202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219	The Job ID Group (MANDATORY)	
202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218	The Job ID Group (MANDATORY)	

Job Monitoring MIB, V0.85 Aug 8, 1997

222 223	6. APPENDIX B - SUPPORT OF THE JOB SUBMISSION ID IN JOB SUBMISSION PROTOCOLS	81
224	6.1 Hewlett-Packard's Printer Job Language (PJL)	82
225	6.2 ISO DPA	82
226	7. REFERENCES	82
227	8. AUTHOR'S ADDRESSES	83
228 229	9. INDEX	87

Job Monitoring MIB

231 **1. Introduction**

230

232 The Job Monitoring MIB is intended to be implemented by an agent within a printer or the 233 first server closest to the printer, where the printer is either directly connected to the 234 server only or the printer does not contain the job monitoring MIB agent. It is 235 recommended that implementations place the SNMP agent as close as possible to the 236 processing of the print job. This MIB applies to printers with and without spooling 237 capabilities. This MIB is designed to be compatible with most current commonly-used job 238 submission protocols. In most environments that support high function job submission/job 239 control protocols, like ISO DPA[iso-dpa], those protocols would be used to monitor and 240 manage print jobs rather than using the Job Monitoring MIB.

The Job Monitoring MIB consists of a General Group, a Job Submission ID Group, a Job Group, and an Attribute Group. Each group is a table. All accessible objects are read-

243 only. The General Group contains general information that applies to all jobs in a job set.

244 The Job Submission ID table maps the job submission ID that the client uses to identify a

job to the **jmJobIndex** that the Job Monitoring Agent uses to identify jobs in the Job and

246 Attribute tables. The Job table contains the MANDATORY integer job state and status

247 objects. The Attribute table consists of multiple entries per job that specify (1) job and

248 document identification and parameters, (2) requested resources, and (3) consumed

249 resources during and after job processing/printing. A larger number of job attributes are

- defined as textual conventions that an agent SHALL return if the server or device
- implements the functionality so represented and the agent has access to the information.

1.1 Types of Information in the MIB

253 The job MIB is intended to provide the following information for the indicated Role

254 Models in the Printer MIB[print-mib] (Appendix D - Roles of Users).

255	User:
256 257 258	Provide the ability to identify the least busy printer. The user will be able to determine the number and size of jobs waiting for each printer. No attempt is made to actually predict the length of time that jobs will take.
259	Provide the ability to identify the current status of the user's job (user queries).
260	Provide a timely indication that the job has completed and where it can be found.
261 262	Provide error and diagnostic information for jobs that did not successfully complete.

263 Operator:

264	Provide a presentation of the state of all the jobs in the print system.
265	Provide the ability to identify the user that submitted the print job.
266	Provide the ability to identify the resources required by each job.
267 268	Provide the ability to define which physical printers are candidates for the print job.
269 270 271	Provide some idea of how long each job will take. However, exact estimates of time to process a job is not being attempted. Instead, objects are included that allow the operator to be able to make gross estimates.
272	Capacity Planner:
273	Provide the ability to determine printer utilization as a function of time.
274	Provide the ability to determine how long jobs wait before starting to print.
275	Accountant:
276 277	Provide information to allow the creation of a record of resources consumed and printer usage data for charging users or groups for resources consumed.
278 279	Provide information to allow the prediction of consumable usage and resource need.
280 281 282 283 284	The MIB supports printers that can contain more than one job at a time, but still be usable for low end printers that only contain a single job at a time. In particular, the MIB supports the needs of Windows and other PC environments for managing low-end direct- connect (serial or parallel) and networked devices without unnecessary overhead or complexity, while also providing for higher end systems and devices.
285	1.2 Types of Job Monitoring Applications
286	The Job Monitoring MIB is designed for the following types of monitoring applications:
287 288 289	1. Monitor a single job starting when the job is submitted and ending a defined period after the job completes. The Job Submission ID table provides the map to find the specific job to be monitored.
290 291 292 293 294 295 296 297	2. Monitor all 'active' jobs in a queue, which this specification generalizes to a "job set". End users may use such a program when selecting a least busy printer, so the MIB is designed for such a program to start up quickly and find the information needed quickly without having to read all (completed) jobs in order to find the active jobs. System operators may also use such a program, in which case it would be running for a long period of time and may also be interested in the jobs that have completed. Finally such a program may be used to provide an enhanced console and logging capability.

- 298 3. Collect resource usage for accounting or system utilization purposes that copy 299 the completed job statistics to an accounting system. It is recognized that 300 depending on accounting programs to copy MIB data during the job-retention 301 period is somewhat unreliable, since the accounting program may not be 302 running (or may have crashed). Such a program is also expected to keep a 303 shadow copy of the entire Job Attribute table including completed, 304 canceled, and aborted jobs which the program updates on each polling cycle. 305 Such a program polls at the rate of the persistence of the Attribute table. 306 The design is not optimized to help such an application determine which jobs 307 are **completed**, **canceled**, or **aborted**. Instead, the application SHALL query 308 each job that the application's shadow copy shows was not complete, 309 canceled, or aborted at the previous poll cycle to see if it is now complete or 310 canceled, plus any new jobs that have been submitted.
- 311 The MIB provides a set of objects that represent a compatible subset of job and document
- 312 attributes of the ISO DPA standard[iso-dpa] and the Internet Printing Protocol (IPP)[ipp-
- 313 model], so that coherence is maintained between these two protocols and the information
- 314 presented to end users and system operators by monitoring applications. However, the
- 315 job monitoring MIB is intended to be used with printers that implement other job
- submitting and management protocols, such as IEEE 1284.1 (TIPSI)[tipsi], as well as
- 317 with ones that do implement ISO DPA. Thus the job monitoring MIB does not require
- 318 implementation of either the ISO DPA or IPP protocols.
- The MIB is designed so that an additional MIB(s) can be specified in the future for monitoring multi-function (scan, FAX, copy) jobs as an augmentation to this MIB.

321 **2. Terminology and Job Model**

- This section defines the terms that are used in this specification and the general model for jobs.
- NOTE Existing systems use conflicting terms, so these terms are drawn from the ISO
- 325 10175 Document Printing Application (DPA) standard[iso-dpa]. For example,
- PostScript systems use the term *session* for what is called a *job* in this specification and
- 327 the term *job* to mean what is called a *document* in this specification. PJL systems use
- 328 the term *job* to mean what is called a *job* in this specification. PJL also supports
- 329 multiple *documents* per job, but does not support specifying per-document attributes
- independently for each document.
- Job: a unit of work whose results are expected together without interjection of unrelatedresults. A job contains one or more *documents*.
- 333 Job Set: a group of jobs that are queued and scheduled together according to a specified
- 334 scheduling algorithm for a specified device or set of devices. For implementations that
- embed the SNMP agent in the device, the MIB job set normally represents *all* the jobs
- known to the device, so that the implementation only implements a single job set. If the

- 337 SNMP agent is implemented in a server that controls one or more devices, each MIB job
- 338 set represents a job queue for (1) a specific device or (2) set of devices, if the server uses a

339 single queue to load balance between several devices. Each job set is disjoint; no job

- 340 SHALL be represented in more than one MIB job set.
- 341 Document: a sub-section within a job that contains print data and *document instructions*342 that apply to just the document.
- 343 Client: the network entity that *end users* use to submit jobs to *spoolers*, *servers*, or
- 344 *printers* and other *devices*, depending on the configuration, using any job submission
- 345 protocol over a serial or parallel port to a directly-connected device or over the network
- to a networked-connected device.
- 347 Server: a network entity that accepts jobs from clients and in turn submits the jobs to
- 348 *printers* and other *devices* that may be directly connected to the server via a serial or
- 349 parallel port or may be on the network. A server MAY be a printer *supervisor* control
- 350 program, or a print *spooler*.
- 351 Device: a hardware entity that (1) interfaces to humans in human perceptible means, such
- as produces marks on paper, scans marks on paper to produce an electronic
- 353 representations, or writes CD-ROMs or (2) interfaces electronically to another device,
- 354 such as sends FAX data to another FAX device.
- 355 Printer: a *device* that puts marks on media.
- Supervisor: a server that contains a control program that controls a printer or otherdevice. A supervisor is a client to the printer or other device.
- 358 Spooler: a server that accepts jobs, spools the data, and decides when and on which
- printer to print the job. A spooler is a client to a printer or a printer supervisor, dependingon implementation.
- Spooling: the act of a *device* or *server* of (1) accepting jobs and (2) writing the job's attributes and document data on to secondary storage.
- 363 Queuing: the act of a *device* or *server* of ordering (queuing) the jobs for the purposes of 364 scheduling the jobs to be processed.
- 365 Monitor or Job Monitoring Application: the SNMP management application that End
- 366 Users, and System Operators use to monitor jobs using SNMP. A monitor MAY be either
- 367 a separate application or MAY be part of the client that also submits jobs.
- 368 Accounting Application: the SNMP management application that copies job information
- 369 to some more permanent medium so that another application can perform accounting on
- the data for Accountants, Asset Managers, and Capacity Planners use.

- 371 Agent: the network entity that accepts SNMP requests from a *monitor* or *accounting*
- 372 *application* and provides access to the instrumentation for managing jobs modeled by the
- 373 management objects defined in the Job Monitoring MIB module for a *server* or a *device*.
- 374 Proxy: an agent that acts as a concentrator for one or more other agents by accepting
- 375 SNMP operations on the behalf of one or more other agents, forwarding them on to those
- other agents, gathering responses from those other agents and returning them to the
- 377 original requesting monitor.
- 378 User: a person that uses a client or a monitor.
- 379 End User: a user that uses a client to submit a print job.
- 380 System Operator: a user that uses a monitor to monitor the system and carries out tasks381 to keep the system running.
- 382 System Administrator: a user that specifies policy for the system.
- 383 Job Instruction: an instruction specifying how, when, or where the job is to be processed.
- 384 Job instructions MAY be passed in the job submission protocol or MAY be embedded in
- 385 the document data or a combination depending on the job submission protocol and 386 implementation.
- 387 Document Instruction: an instruction specifying how to process the document.
- 388 Document instructions MAY be passed in the job submission protocol separate from the
- actual document data, or MAY be embedded in the document data or a combination,
- 390 depending on the job submission protocol and implementation.
- 391 SNMP Information Object: a name, value-pair that specifies an action, a status, or a
- 392 condition in an SNMP MIB. Objects are identified in SNMP by an OBJECT
- 393 IDENTIFIER.
- Attribute: a name, value-pair that specifies a job or document instruction, a status, or a
- 395 condition of a job or a document that has been submitted to a server or device. A
- 396 particular attribute NEED NOT be present in each job instance. In other words, attributes
- 397 are present in a job instance only when there is a need to express the value, either because
- 398 (1) the client supplied a value in the job submission protocol, (2) the document data
- 399 contained an embedded attribute, or (3) the server or device supplied a default value. An
- 400 agent SHALL represent an attribute as an entry (row) in the Attribute table in this MIB in
- 401 which entries are present only when necessary. Attributes are identified in this MIB by an
- 402 enum.
- 403 Job Monitoring (using SNMP): the activity of a management application of accessing the
- 404 MIB and (1) identifying jobs in the job tables being processed by the server, printer or
- 405 other devices, and (2) displaying information to the user about the processing of the job.

406 Job Accounting: the activity of a management application of accessing the MIB and 407 recording what happens to the job during and after the processing of the job.

408 **2.1 System Configurations for the Job Monitoring MIB**

409 This section enumerates the three configurations in which the Job Monitoring MIB is

410 intended to be used. To simplify the pictures, the *devices* are shown as *printers*. See411 section 1.1 entitled "Types of Information in the MIB".

412 The diagram in the Printer MIB[print-mib] entitled: "One Printer's View of the Network"

413 is assumed for this MIB as well. Please refer to that diagram to aid in understanding the414 following system configurations.

415 **2.1.1 Configuration 1 - client-printer**

423

442

416 In the **client-printer** configuration 1, the **client**(s) submit jobs directly to the **printer**,

417 either by some direct connect, or by network connection.

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

+23				
424	all	end-user	########	SNMP query
425	++	++	job	submission
426	monitor	client		
427	++	+#++		
428	#	#		
429	# #####	+######		
430	# #			
431	+==+===#=#=+==+	-		
432	agent			
433	++			
434	PRINTER <	<+		
435		Print Job Del:	ivery Chann	nel
436				
437	+================+	-		

438 **Figure 2-1 - Configuration 1 - client-printer - agent in the printer**

The Job Monitoring MIB is designed to support the following relationships (not shown inFigure 2-1):

- 1. Multiple **clients** MAY submit jobs to a **printer**.
 - 2. Multiple clients MAY monitor a printer.

- 443 3. Multiple **monitors** MAY monitor a **printer**.
- 444 4. A **client** MAY submit jobs to multiple **printers**.
- 445 5. A **monitor** MAY monitor multiple **printers**.

446 **2.1.2** Configuration 2 - client-server-printer - agent in the server

447 In the **client-server-printer** configuration 2, the **client**(s) submit jobs to an intermediate

448 server by some network connection, *not* directly to the printer. While configuration 2 is

- included, the design center for this MIB is configurations 1 and 3.
- 450 The job submitting client and/or monitoring application monitor jobs by communicating451 directly with:
- 452 A Job Monitoring MIB agent that is part of the **server** (or a front for the server)

453 There is no SNMP Job Monitoring MIB agent in the **printer** in configuration 2, at least

that the client or monitor are aware. In this configuration, the agent SHALL return the

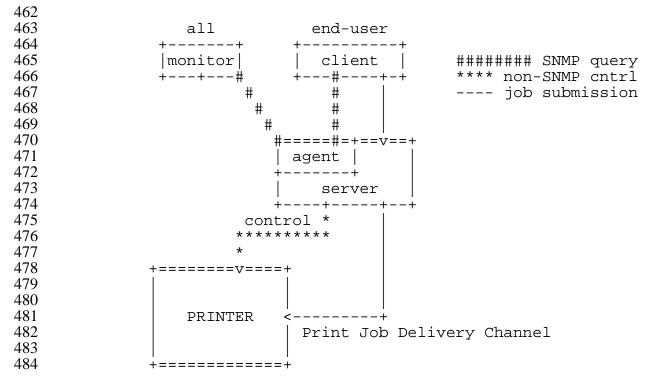
455 current values of the objects in the Job Monitoring MIB both for jobs the server keeps and

456 jobs that the server has submitted to the **printer**. The Job Monitoring MIB agent SHALL

457 obtain the required information from the **printer** by a method that is beyond the scope of

- this document. The agent in the server SHALL keep the job in the Job Monitoring MIB
- in the server as long as the job is in the **printer**, plus a defined time period after the job
- 460 enters the **completed** state in which accounting programs can copy out the accounting

461 data from the Job Monitoring MIB.



485 Figure 2-2 - Configuration 2 - client-server-printer - agent in the server

486 The Job Monitoring MIB is designed to support the following relationships (not shown in Figure 2-2): 487

- 488 1. Multiple **clients** MAY submit jobs to a **server**.
- 489
- Multiple clients MAY monitor a server. 2.
- 490 3. Multiple monitors MAY monitor a server.
- 491 4. A **client** MAY submit jobs to multiple **servers**.
- 492 5. A monitor MAY monitor multiple servers.
- 493 Multiple servers MAY submit jobs to a printer. 6.
- 494 Multiple servers MAY control a printer. 7.

495 2.1.3 Configuration 3 - client-server-printer - client monitors printer agent and 496 server

- 497 In the **client-server-printer** configuration 3, the **client**(s) submit jobs to an intermediate
- 498 server by some network connection, *not* directly to the **printer**. That server does *not* 499 contain a Job Monitoring MIB agent.
- 500 The job submitting **client** and/or **monitoring application** monitor jobs by communicating 501 directly with:
- 502 1. The server using some undefined protocol to monitor jobs in the server (that 503 does not contain the Job Monitoring MIB) AND

504 505 506

507

508

545

546

547

2. A Job Monitoring MIB agent that is part of the **printer** to monitor jobs after the **server** passes the jobs to the **printer**. In such configurations, the **server** deletes its copy of the job from the **server** after submitting the job to the printer usually almost immediately (before the job does much processing, if any).

509 In configuration 3, the agent (in the **printer**) SHALL keep the values of the objects in the 510 Job Monitoring MIB that the agent implements updated for a job that the server has 511 submitted to the printer. The agent SHALL obtain information about the jobs submitted 512 to the printer from the server (either in the job submission protocol, in the document data, 513 or by direct query of the server), in order to populate some of the objects the Job 514 Monitoring MIB in the printer. The agent in the printer SHALL keep the job in the Job 515 Monitoring MIB as long as the job is in the Printer, and longer in order to implement the 516 completed state in which monitoring programs can copy out the accounting data from the 517 Job Monitoring MIB.

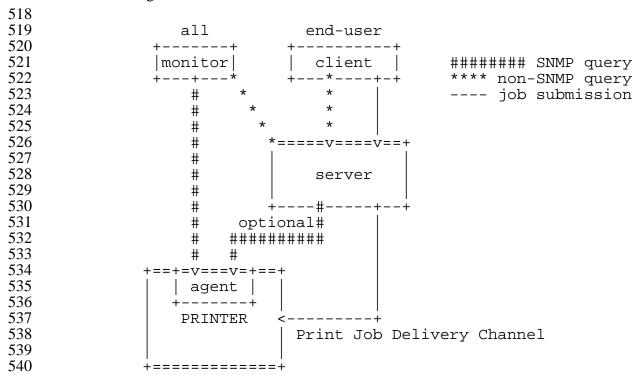


Figure 2-3 - Configuration 3 - client-server-printer - client monitors printer agent and server

- 543 The Job Monitoring MIB is designed to support the following relationships (not shown in544 Figure 2-3):
 - 1. Multiple **clients** MAY submit jobs to a **server**.
 - 2. Multiple clients MAY monitor a server.
 - 3. Multiple **monitors** MAY monitor a **server**.

- 548 4. A **client** MAY submit jobs to multiple **servers**.
- 549 5. A **monitor** MAY monitor multiple **servers**.
- 550 6. Multiple servers MAY submit jobs to a printer.
- 551 7. Multiple servers MAY control a printer.

552 **3. Managed Object Usage**

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

554 **3.1 Conformance Considerations**

- 555 In order to achieve interoperability between job monitoring applications and job
- 556 monitoring agents, this specification includes the conformance requirements for both
- 557 monitoring applications and agents.

558 **3.1.1 Conformance Terminology**

559 This specification uses the verbs: "SHALL", "SHOULD", "MAY", and "NEED NOT" to 560 specify conformance requirements according to RFC 2119 [req-words] as follows:

- SHALL": indicates an action that the subject of the sentence must implement in order to claim conformance to this specification
- * "MAY": indicates an action that the subject of the sentence does not have to
 implement in order to claim conformance to this specification, in other words that
 action is an implementation option
- * "NEED NOT": indicates an action that the subject of the sentence does not have to implement in order to claim conformance to this specification. The verb "NEED NOT" is used instead of "may not", since "may not" sounds like a prohibition.
- "SHOULD": indicates an action that is recommended for the subject of the
 sentence to implement, but is not required, in order to claim conformance to this
 specification.

572 **3.1.2 Agent Conformance Requirements**

- 573 A conforming agent:
- 574 1. SHALL implement *all* MANDATORY groups in this specification.
- 575
 576
 576
 577
 2. SHALL implement any attributes if (1) the server or device supports the functionality represented by the attribute and (2) the information is available to the agent.
- 578
 579
 3. SHOULD implement both forms of an attribute if it implements an attribute that permits a choice of INTEGER and OCTET STRING forms, since

- implementing both forms may help management applications by giving them a
 choice of representations, since the representation are equivalent. See the
 JmAttributeTypeTC textual-convention.
- 583 NOTE This MIB, like the Printer MIB, is written following the subset of SMIv2 that
 584 can be supported by SMIv1 and SNMPv1 implementations.
- 585 3.1.2.1 MIB II System Group objects
- 586 The Job Monitoring MIB agent SHALL implement all objects in the System Group of
- 587 MIB-II[mib-II], whether the Printer MIB[print-mib] is implemented or not.
- 588 3.1.2.2 MIB II Interface Group objects
- 589 The Job Monitoring MIB agent SHALL implement all objects in the Interfaces Group of
- 590 MIB-II[mib-II], whether the Printer MIB[print-mib] is implemented or not.
- 591 3.1.2.3 Printer MIB objects
- 592 If the agent is providing access to a device that is a printer, the agent SHALL implement
- all of the MANDATORY objects in the Printer MIB[print-mib] and all the objects in other
- 594 MIBs that conformance to the Printer MIB requires, such as the Host Resources MIB[hr-
- 595 mib]. If the agent is providing access to a server that controls one or more direct-connect
- 596 or networked printers, the agent NEED NOT implement the Printer MIB and NEED NOT
- 597 implement the Host Resources MIB.

598 **3.1.3** Job Monitoring Application Conformance Requirements

599 A conforming job monitoring application:

600 601 602 603	1.	SHALL accept the full syntactic range for all objects in all MANDATORY groups and all MANDATORY attributes that are required to be implemented by an agent according to Section 3.1.2 and SHALL either present them to the user or ignore them.
604 605 606 607 608 609	2.	SHALL accept the full syntactic range for <i>all</i> attributes, including enum and bit values specified in this specification and additional ones that may be registered with IANA and SHALL either present them to the user or ignore them. In particular, a conforming job monitoring application SHALL not malfunction when receiving any standard or registered enum or bit values. See Section 3.6 entitled "IANA Considerations".
610 611	3.	SHALL NOT fail when operating with agents that materialize attributes <i>after</i> the job has been submitted, as opposed to when the job is submitted.
612 613	4.	SHALL, if it supports a time attribute, accept either form of the time attribute, since agents are free to implement either time form.

614	3.2 The Job Tables and the Oldest Active and Newest Active Indexes
615 616	The jmJobTable and jmAttributeTable contain objects and attributes, respectively, for each job in a job set. These first two indexes are:
617 618	 jmGeneralJobSetIndex - which job set jmJobIndex - which job in the job set
619 620	In order for a monitoring application to quickly find that active jobs (jobs in the pending , processing , or processingStopped states), the MIB contains two indexes:
621 622	1. jmGeneralOldestActiveJobIndex - the index of the active job that has been in the tables the longest.
623 624	2. jmGeneralNewestActiveJobIndex - the index of the active job that has been most recently added to the tables.
625 626 627 628 629 630	The agent SHALL assign the next incremental value of jmJobIndex to the job, when a new job is accepted by the server or device to which the agent is providing access. If the incremented value of jmJobIndex would exceed the implementation-defined maximum value for jmJobIndex , the agent SHALL 'wrap' back to 1. An agent uses the resulting value of jmJobIndex for storing information in the jmJobTable and the jmAttributeTable about the job.
631 632 633 634	It is recommended that the largest value for jmJobIndex be much larger than the maximum number of jobs that the implementation can contain at a single time, so as to minimize the premature re-use of a jmJobIndex value for a newer job while clients retain the same 'stale' value for an older job.
635 636 637 638 639 640 641 642 643	It is recommended that agents that are providing access to servers/devices that already allocate job-identifiers for jobs as integers use the same integer value for the jmJobIndex . Then the jobs will have the same job identifier value as the jmJobIndex value, so that users viewing jobs by management applications using this MIB and applications using other protocols will see the same job identifiers for the same jobs. Agents providing access to systems that contain jobs with a job identifier of 0 SHALL map the job identifier value 0 to a jmJobIndex value that is one higher than the highest job identifier value that any job can have on that system. Then only job 0 will have a different job-identifier value that he job's jmJobIndex value.
644 645 646	NOTE - If a server or device accepts jobs using multiple job submission protocols, it may be difficult for the agent to meet the recommendation to use the job-identifier values that the server or device assigns as the jmJobIndex value, unless the server/device assigns

- job-identifiers for each of its job submission protocols from the same job-identifier number
- 648 space.
- Each time a new job is accepted by the server or device that the agent is providing access
- to AND that job is to be 'active' (**pending**, **processing**, or **processingStopped**, but not
- 651 **pendingHeld**), the agent SHALL copy the value of the job's **jmJobIndex** to the

- 652 **jmGeneralNewestActiveJobIndex** object. If the new job is to be 'inactive'
- 653 (**pendingHeld** state), the agent SHALL not change the value of
- 654 **jmGeneralNewestActiveJobIndex** object (though the agent SHALL assign the next
- 655 incremental **jmJobIndex** value to the job).
- 656 When a job transitions from one of the 'active' job states (pending, processing,
- 657 **processingStopped**) to one of the 'inactive' job states (**pendingHeld**, **completed**,
- 658 canceled, or aborted), with a jmJobIndex value that matches the
- 659 **jmGeneralOldestActiveJobIndex** object, the agent SHALL advance (or wrap) the value
- 660 to the next oldest 'active' job, if any. See the **JmJobStateTC** textual-convention for a 661 definition of the job states.
- 662 Whenever a job transitions from one of the 'inactive' job states to one of the 'active' job
- states (from **pendingHeld** to **pending** or **processing**), the agent SHALL update the value
- of either the **jmGeneralOldestActiveJobIndex** or the
- 665 **jmGeneralNewestActiveJobIndex** objects, or both, if the job's **jmJobIndex** value is
- outside the range between **jmGeneralOldestActiveJobIndex** and
- 667 **jmGeneralNewestActiveJobIndex**.
- 668 When all jobs become 'inactive', i.e., enter the **pendingHeld**, **completed**, **canceled**, or
- **aborted** states, the agent SHALL set the value of both the
- 670 **jmGeneralOldestActiveJobIndex** and **jmGeneralNewestActiveJobIndex** objects to **0**.
- 671 NOTE Applications that wish to efficiently access all of the active jobs MAY use
- 672 **jmGeneralOldestActiveJobIndex** value to start with the oldest active job and continue
- 673 until they reach the index value equal to **jmGeneralNewestActiveJobIndex**, skipping
- over any **pendingHeld**, **completed**, **canceled**, **or aborted** jobs that might intervene.
- 675 If an application detects that the **jmGeneralNewestActiveJobIndex** is smaller than
- 676 **jmGeneralOldestActiveJobIndex**, the job index has wrapped. In this case, the
- application SHALL reset the index to **1** when the end of the table is reached and continue
- 678 the GetNext operations to find the rest of the active jobs.
- 679 NOTE Application detect the end of the **jmAttributeTable** table when the OID
- returned by the GetNext operation is an OID in a different MIB. There is no object in this
- 681 MIB that specifies the maximum value for the **jmJobIndex** supported by the
- 682 implementation.
- 683 When the server or device is power-cycled, the agent SHALL remember the next
- 684 **jmJobIndex** value to be assigned, so that new jobs are not assigned the same
- 685 **jmJobIndex** as recent jobs before the power cycle.

686 **3.3 The Attribute Mechanism**

- 687 Attributes are similar to information objects, except that attributes are identified by an
- enum, instead of an OID, so that attributes may be registered without requiring a new
- 689 MIB. Also an implementation that does not have the functionality represented by the 690 attribute can omit the attribute entirely, rather than having to return a distinguished value
- attribute can omit the attribute entirely, rather than having to return a distinguished value.
 The agent is free to materialize an attribute in the **imAttributeTable** as soon as the agent
- 692 is aware of the value of the attribute.
- 693 The agent materializes job attributes in a four-indexed **jmAttributeTable**:
- 694 1. jmGeneralJobSetIndex which job set
- 695 2. jmJobIndex which job in the job set
- 696 3. jmAttributeTypeIndex which attribute
- 6976984. jmAttributeInstanceIndex which attribute instance for those attributes that can have multiple values per job.
- 699 Some attributes represent information about a job, such as a file-name, a document-name,
- a submission-time or a completion time. Other attributes represent resources required,
- e.g., a medium or a colorant, etc. to process the job before the job starts processing OR to
- indicate the amount of the resource consumed during and after processing, e.g., pages
- completed or impressions completed. If both a required and a consumed value of a
- resource is needed, this specification assigns two separate attribute enums in the textual convention.
- NOTE The table of contents lists all the attributes in order. This order is the order of
- enum assignments which is the order that the SNMP GetNext operation returns attributes.

708 Most attributes apply to all three configurations covered by this MIB specification (see

- section 2.1 entitled "System Configurations for the Job Monitoring MIB"). Those
- 710 attributes that apply to a particular configuration are indicated as 'Configuration *n*:' and
- 711 SHALL NOT be used with other configurations.

712 **3.3.1 Conformance of Attribute Implementation**

- An agent SHALL implement any attribute if (1) the server or device supports the
- functionality represented by the attribute and (2) the information is available to the agent.
- 715 The agent MAY create the attribute row in the **jmAttributeTable** when the information is
- available or MAY create the row earlier with the designated 'unknown' value appropriate
- 717 for that attribute. See next section.
- 718 If the server or device does not implement or does not provide access to the information
- about an attribute, the agent SHOULD NOT create the corresponding row in the
- 720 **jmAttributeTable**.

721 **3.3.2** Useful, 'Unknown', and 'Other' Values for Objects and Attributes

Some attributes have a 'useful' Integer32 value, some have a 'useful' OCTET STRING

value, some MAY have either or both depending on implementation, and some MUST

- have both. See the JmAttributeTypeTC textual convention for the specification of eachattribute.
- SNMP requires that if an object cannot be implemented because its values cannot be
- accessed, then a compliant agent SHALL return an SNMP error in SNMPv1 or an
- exception value in SNMPv2. However, this MIB has been designed so that 'all' objects
- can and SHALL be implemented by an agent, so that neither the SNMPv1 error nor the
- 730 SNMPv2 exception value SHALL be generated by the agent. This MIB has also been
- designed so that when an agent materializes an attribute, the agent SHALL materialize a
- row consisting of both the **jmAttributeValueAsInteger** and **jmAttributeValueAsOctets**
- 733 objects.

In general, values for objects and attributes have been chosen so that a management

application will be able to determine whether a 'useful', 'unknown', or 'other' value is

available. When a useful value is not available for an object that agent SHALL return a

737 zero-length string for octet strings, the value 'unknown(2)' for enums, a '0' value for an

- 738 object that represents an index in another table, and a value '-2' for counting integers.
- 739 Since each attribute is represented by a row consisting of both the

740 jmAttributeValueAsInteger and jmAttributeValueAsOctets MANDATORY objects,

- 741 SNMP requires that the agent SHALL always create an attribute row with both objects
- specified. However, for most attributes the agent SHALL return a "useful" value for one
- of the objects and SHALL return the 'other' value for the other object. For integer only
- attributes, the agent SHALL always return a zero-length string value for the
- 745 **jmAttributeValueAsOctets** object. For octet string only attributes, the agent SHALL
- always return a '-1' value for the **jmAttributeValueAsInteger** object.

747 **3.3.3 Data Sub-types and Attribute Naming Conventions**

Many attributes are sub-typed to give a more specific data type than Integer32 or OCTET STRING. The data sub-type of each attribute is indicated on the first line(s) of the description. Some attributes have several different data sub-type representations.
When an attribute has both an Integer32 data sub-type and an OCTET STRING data sub-type, the attribute can be represented in a single row in the jmAttributeTable. In this case, the data sub-type name is not included as the last part of the name of the attribute, e.g., documentFormat(38) which is both an enum and/or a name. When the

- data sub-types cannot be represented by a single row in the **jmAttributeTable**, each such
- representation is considered a separate attribute and is assigned a separate name and enum
- value. For these attributes, the name of the data sub-type is the last part of the name of

- the attribute: Name, Index, DateAndTime, TimeStamp, etc. For example,
- 759 **documentFormatIndex(37)** is an index.
- 760 NOTE: The Table of Contents also lists the data sub-type and/or data sub-types of each
- attribute, using the textual-convention name when such is defined. The following
- abbreviations are used in the Table of Contents as shown:

	L_{1}
'Int32(-2)'	Integer32(-22147483647)
'Int32(0)'	Integer32(02147483647)
'Int32(1)'	Integer32(12147483647)
'Int32(mn)'	For all other Integer ranges, the lower and upper bound of
	the range is indicated.
'UTF8String63'	JmUTF8StringTC(SIZE(063))
'JobString63'	JmJobStringTC(SIZE(063))
'Octets63'	OCTET STRING(SIZE(063))
'Octets(mn)'	For all other OCTET STRING ranges, the exact range is
	indicated.

763 3.3.4 Single-Value (Row) Versus Multi-Value (MULTI-ROW) Attributes

- 764 Most attributes SHALL have only one row per job. However, a few attributes can have
- multiple values per job or even per document, where each value is a separate row in the
- 766 jmAttributeTable. Unless indicated with 'MULTI-ROW:' in the JmAttributeTypeTC
- description, an agent SHALL ensure that each attribute occurs only once in the
- 768 **jmAttributeTable** for a job. Most of the '**MULTI-ROW**' attributes do not allow
- duplicate values, i.e., the agent SHALL ensure that each value occurs only once for a job.
- 770 Only if the specification of the 'MULTI-ROW' attribute also says "the values NEED NOT
- be unique" can the agent allow duplicate values to occur for the job.
- 772 NOTE Duplicates are allowed for 'extensive' 'MULTI-ROW' attributes, such as
- fileName(34) or documentName(35) which are specified to be 'per-document' attributes,
- but are *not* allowed for 'intensive' '**MULTI-ROW**' attributes, such as
- 775 mediumConsumed(171) and documentFormat(38) which are specified to be 'per-job'
- attributes.

777 **3.3.5 Requested Attributes**

- A number of attributes record requirements for the job. Such attribute names end with the
- word '**Requested**'. In the interests of brevity, the phrase 'requested' SHALL mean: (1)
- requested by the client (or intervening server) in the job submission protocol and MAY
- also mean (2) embedded in the submitted document data, and/or (3) defaulted by the
- recipient device or server with the same semantics as if the requester had supplied,
- 783 depending on implementation.

784 **3.3.6 Consumption Attributes**

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

791 **3.3.7 Index Value Attributes**

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

798 **3.4 Job Identification**

799 There are a number of attributes that permit a user, operator or system administrator to 800 identify jobs of interest, such as **jobName**, **jobOriginatingHost**, etc. In addition, there is 801 a **jmJobSubmissionID** object that is a text string table index. Being a table index allows 802 a monitoring application to quickly locate and identify a particular job of interest that was submitted from a particular client by the user invoking the monitoring application. The 803 804 Job Monitoring MIB needs to provide for identification of the job at both sides of the job 805 submission process. The primary identification point is the client side. The jmJobSubmissionID allows the monitoring application to identify the job of interest from 806 807 all the jobs currently "known" by the server or device. The value of jmJobSubmissionID 808 can be assigned by either the client's local system or a downstream server or device. The 809 point of assignment depends on the job submission protocol in use. 810 The server/device-side identifier, called the **imJobIndex** object, SHALL be assigned by

- 811 the SNMP Job Monitoring MIB agent when the server or device accepts the jobs from
- 812 submitting clients. The **jmJobIndex** object allows the interested party to obtain all
- 813 objects desired that relate to a particular job. See Section 3.2, entitled 'The Job Tables
- and the Oldest Active and Newest Active Indexes' for the specification of how the agent
- 815 shall assign the **jmJobIndex** values.
- 816 NOTE For a number of job submission protocols the server/device assigns an integer job
- 817 identifier when accepting a job so that the submitting client can reference the job in
- 818 subsequent protocol operations (For example, see IPP [ipp]). For such implementations,

- 819 it is recommended that the value of the job identifier and the value of jmJobIndex be the820 same, so that
- 821 The MIB provides a mapping table that maps each **jmJobSubmissionID** value to the
- 822 corresponding **jmJobIndex** value generated by the agent, so that an application can
- 823 determine the correct value for the **jmJobIndex** value for the job of interest in a single
- 624 Get operation, given the Job Submission ID. See the **jmJobIDGroup**.
- 825 The **jobName** attribute provides a name that the user supplies as a job attribute with the
- job. The jobName attribute is not necessarily unique, even for one user, let alone acrossusers.
- 828 **3.5 Internationalization Considerations**
- 829 This section describes the internationalization considerations included in this MIB.

830 **3.5.1** 'JmUTF8StringTC' for text generated by the server or device

- 831 There are a few objects and attributes that are represented using the Universal Multiple-
- 832 Octet Coded Character Set (UCS) [ISO-10646] encoded as an octet string using the UTF-
- 833 8 [UTF-8] character encoding scheme. The '**JmUTF8StringTC**' textual convention is
- used to indicate UTF-8 text strings. These objects and attributes are always supplied (if
 implemented) by the agent, not by the job submitting client:
- 836 1.

837

838

- jmGeneralJobSetName object
 processingMessage(6) attribute
- 3. physicalDevice(32) (name value) attribute
- 839 The coded character set for representing these objects and attributes SHALL be UTF-8 as
- recommended by RFC 2130 [RFC 2130] and the "IETF Policy on Character Sets andLanguage" [char-set policy].
- NOTE For strings in 7-bit US-ASCII, there is no impact since the UTF-8 representation
 is identical to the US-ASCII [US-ASCII] encoding.

844 **3.5.2** 'JmJobStringTC' for text generated by the job submitter

845 All of the objects and attributes represented by the 'JmJobStringTC' textual-convention

- are either (1) supplied in the job submission protocol by the client that submits the job to
- the server or device or (2) are defaulted by the server or device if the job submitting client
- does not supply values. The agent SHALL represent these objects and attributes in the
- 849 MIB either (1) in the coded character set as they were submitted or (2) MAY convert the
- 850 coded character set to another coded character set or encoding scheme. In any case, the
- resulting coded character set representation SHOULD be UTF-8 [UTF-8], but SHALL be
- one in which the code positions from 0 to 31 SHALL not be used, 32 to 127 SHALL be

US-ASCII [US-ASCII], 127 SHALL be unused, and the remaining code positions 128 to 255 SHALL represent single-byte or multi-byte graphic characters structured according to

855 ISO 2022 [ISO 2022] or SHALL be unused.

856 The coded character set SHALL be one of the ones registered with IANA [IANA] and

857 SHALL be identified by the **jobCodedCharSet** attribute in the **jmJobAttributeTable** for

the job. If the agent does not know what coded character set was used by the job

submitting client, the agent SHALL return the '**unknown**(2)' value for the

860 **jobCodedCharSet** attribute for the job.

861 Examples of coded character sets which meet this criteria for use as the value of the

jobCodedCharSet job attribute are: US-ASCII [US-ASCII], ISO 8859-1 (Latin-1) [ISO

863 8859-1], any ISO 8859-n, HP Roman8, IBM Code Page 850, Windows Default 8-bit set,

864 UTF-8 [UTF-8], US-ASCII plus JIS X0208-1990 Japanese [JIS X0208], US-ASCII plus

- 65 GB2312-1980 PRC Chinese [GB2312]. See the IANA registry of coded character sets
- 866 [IANA charsets].

867 Examples of coded character sets which do not meet this criteria are: national 7-bit sets

conforming to ISO 646 (except US-ASCII), EBCDIC, and ISO 10646 (Unicode) [ISO-

869 10646]. In order to represent Unicode characters, the UTF-8 [UTF-8] encoding scheme

870 SHALL be used which has been assigned the MIBenum value of '106' by IANA.

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

873 **3.5.3 'DateAndTime' for representing the date and time**

874 This MIB also contains objects that are represented using the **DateAndTime** textual

convention from SMIv2 [SMIv2-TC]. The job management application SHALL display
such objects in the locale of the user running the monitoring application.

877 **3.6 IANA Considerations**

878 During the development of this standard, the Printer Working Group (PWG) working with 879 IANA [iana] will register additional enums while the standard is in the proposed and draft

states according to the procedures described in this section. IANA will handle registration

of additional enums after this standard is approved in cooperation with an IANA-

appointed registration editor from the PWG according to the procedures described in thissection:

884 **3.6.1 IANA Registration of enums**

This specification uses textual conventions to define enumerated values (enums) and bit
values. Enumerations (enums) and bit values are sets of symbolic values defined for use

with one or more objects or attributes. All enumeration sets and bit value sets are
assigned a symbolic data type name (textual convention). As a convention the symbolic
name ends in "TC" for textual convention. These enumerations are defined at the
beginning of the MIB module specification.

891 This working group has defined several type of enumerations for use in the Job

892 Monitoring MIB and the Printer MIB[print-mib]. These types differ in the method

893 employed to control the addition of new enumerations. Throughout this document,

references to "type n enum", where n can be 1, 2 or 3 can be found in the various tables.

- 895 The definitions of these types of enumerations are:
- 896 3.6.1.1 Type 1 enumerations

897 Type 1 enumeration: All the values are defined in the Job Monitoring MIB specification

- 898 (RFC for the Job Monitoring MIB). Additional enumerated values require a new RFC.
- 899 There are no type 1 enums in the current draft.
- 900 3.6.1.2 Type 2 enumerations
- 901 Type 2 enumeration: An initial set of values are defined in the Job Monitoring MIB

902 specification. Additional enumerated values are registered after review by this working

- group or an editor appointed by IANA after this working group is no longer active.
- 904 The following type 2 enums are contained in the current draft :
- 905 1. JmUTF8StringTC
- 906 2. JmJobStringTC

907 908

909

910 911

912

913

- 3. JmTimeStampTC
 - 4. JmFinishingTC [same enum values as IPP "finishing" attribute]
 - 5. JmPrintQualityTC [same enum values as IPP "print-quality" attribute]
- 6. JmTonerEconomyTC
 - 7. JmMediumTypeŤC
 - 8. JmJobSubmissionTypeTC
 - 9. JmJobStateTC [same enum values as IPP "job-state" attribute]
- 914 10. JmAttributeTypeTC

915 For those textual conventions that have the same enum values as the indicated IPP Job

916 attribute SHALL be simultaneously registered by IANA for use with IPP [ipp-model] and

- 917 the Job Monitoring MIB.
- 918 3.6.1.3 Type 3 enumeration
- 919 Type 3 enumeration: An initial set of values are defined in the Job Monitoring MIB
- 920 specification. Additional enumerated values are registered through IANA without
- 921 working group review.
- 922 There are no type 3 enums in the current draft.

923 **3.6.2 IANA Registration of type 2 bit values**

- 924 This draft contains the following type 2 bit value textual-conventions:
- 925 1. JmJobServiceTypesTC
- 926 2. JmJobStateReasons1TC
- 927 3. JmJobStateReasons2TC
- 928 4. JmJobStateReasons3TC
- 929 5. JmJobStateReasons4TC
- 930 These textual-conventions are defined as bits in an Integer so that they can be used with
- 931 SNMPv1 SMI. The jobStateReasonsN (N=1..4) attributes are defined as bit values using
- 932 the corresponding **JmJobStateReasonsNTC** textual-conventions.
- 933 The registration of JmJobServiceTypesTC and JmJobStateReasonsNTC bit values
- 934 SHALL follow the procedures for a type 2 enum as specified in Section 3.6.1.2.

935 **3.6.3 IANA Registration of Job Submission Id Formats**

- 936 In addition to enums and bit values, this specification assigns a single ASCII digit or letter
- 937 to various job submission ID formats. See the JmJobSubmissionIDTypeTC textual-
- 938 convention and the object. The registration of **jmJobSubmissionID** format numbers
- 939 SHALL follow the procedures for a type 2 enum as specified in Section 3.6.1.2.

940 **3.6.4 IANA Registration of MIME types/sub-types for document-formats**

- 941 The **documentFormat(38)** attribute has MIME type/sub-type values for indicating
- 942 document formats which IANA registers as "media type" names. The values of the
- 943 **documentFormat(38)** attribute are the same as the corresponding Internet Printing
- 944 Protocol (IPP) "document-format" Job attribute values [ipp-model].
- 945 **3.7 Security Considerations**

946 **3.7.1 Read-Write objects**

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

952 3.7.2 Read-Only Objects In Other User's Jobs

The security policy of some sites MAY be that unprivileged users can only get the objects from jobs that they submitted, plus a few minimal objects from other jobs, such as the

955 jmJobKOctetsRequested and jmJobKOctetsProcessed objects, so that a user can tell 956 how busy a printer is. Other sites MAY allow all unprivileged users to see all objects of 957 all jobs. This MIB does not require, nor does it specify how, such restrictions would be 958 implemented. A monitoring application SHOULD enforce the site security policy with

respect to returning information to an unprivileged end user that is using the monitoring

application to monitor jobs that do not belong to that user, i.e., the **jmJobOwner** object

- 961 in the **jmJobTable** does not match the user's user name.
- An operator is a privileged user that would be able to see all objects of all jobs,
- 963 independent of the policy for unprivileged users.

964 **3.8 Notifications**

- 965 This MIB does not specify any notifications. For simplicity, management applications are
- 966 expected to poll for status. The **jmGeneralJobPersistence** and
- 967 jmGeneralAttributePersistence objects assist an application to determine the polling
- 968 rate. The resulting network traffic is not expected to be significant.

969 4. MIB specification

970 The following pages constitute the actual Job Monitoring MIB.

971 972	Job-Monitoring-MIB DEFINITIONS ::= BEGIN	
973	IMPORTS	
	MODULE-IDENTITY, OBJECT-TYPE, experimental, Inte	
		FROM SNMPv2-SMI
	TEXTUAL-CONVENTION MODULE COMPLIANCE OPJECT CROUP	FROM SNMPv2-TC
	MODULE-COMPLIANCE, OBJECT-GROUP The following textual-conventions are needed	FROM SNMPv2-CONF;
	to implement certain attributes, but are <i>not</i>	
	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
974	Coucuenarset	
975	Use the experimental (54) OID assigned to the Printer MIB[print	-mib]
976	before it was published as RFC 1759.	-
977	Upon publication of the Job Monitoring MIB as an RFC, delete t	his
978	comment and the line following this comment and change the	
979	reference of { temp 105 } (below) to { mib-2 X }.	
980 981	This will result in changing: 1 3 6 1 3 54 jobmonMIB(105) to:	
982	1 3 6 1 2 1 jobmonMIB(X)	
983	This will make it easier to translate prototypes to	
984	the standard namespace because the lengths of the OIDs won't	
985	change.	
986 087	temp OBJECT IDENTIFIER ::= { experimental 54 }	
987 988	jobmonMIB MODULE-IDENTITY	
989	LAST-UPDATED "9708080000Z"	
990	ORGANIZATION "IETF Printer MIB Working Group"	
991	CONTACT-INFO	
992	"Tom Hastings	
993	Postal: Xerox Corp.	
994 995	Mail stop ESAE-231 701 S. Aviation Blvd.	
995 996	El Segundo, CA 90245	
997	Li begundo, eri yoz is	
998	Tel: (301)333-6413	
999	Fax: (301)333-5514	
1000	E-mail: hastings@cp10.es.xerox.com	
1001 1002	Sand comments to the printmih WC using the Job Mon	itoring
1002	Send comments to the printmib WG using the Job Mon Project (JMP) Mailing List: jmp@pwg.org	normg
1003	rojeet (init) maning Else. Jup e p. 5.015	
1005	To learn how to subscribe to the JMP mailing list,	
1006	send email to: jmp-request@pwg.org	
1007		

1008 1009 1010	For further information, access the PWG web page under 'JMP': http://www.pwg.org/" DESCRIPTION
1010	"The MIB module for monitoring job in servers, printers, and other devices.
1012	
1013	File: draft-ietf-printmib-job-monitor-05.txt
1014 1015	Version: 0.85"
1015	$::= \{ \text{ temp 105 } \}$
1010	
1018	
1019	Textual conventions for this MIB module
1020	
1021	
1022	
1023	JmUTF8StringTC ::= TEXTUAL-CONVENTION
1024 1025	DISPLAY-HINT "255a" STATUS current
1025	DESCRIPTION
1027	"To facilitate internationalization, this TC represents information taken from the ISO/IEC IS
1028	10646-1 character set, encoded as an octet string using the UTF-8 character encoding scheme.
1029	
1030	NOTE - The values of objects and attributes using this textual convention are generated by the
1031	server or the device, not by the job submitter."
1032 1033	REFERENCE "See section 3.5.1, "JmUTF8StringTC' for text generated by the server or device'."
1033	SYNTAX OCTET STRING (SIZE (063))
1035	
1036	
1037	
1038	
1039	JmJobStringTC ::= TEXTUAL-CONVENTION
1040 1041	STATUS current DESCRIPTION
1041	"To facilitate internationalization, this TC represents information using any coded character set
1043	registered by IANA that has the following properties: (1) code positions from 0 to 31 SHALL
1044	not be used, (2) 32 to 127 SHALL be US-ASCII [US-ASCII], (3) 127 SHALL be unused, and
1045	(4) the remaining code positions 128 to 255 SHALL represent single-byte or multi-byte graphic
1046	characters structured according to ISO 2022 [ISO 2022] or SHALL be unused. While it is
1047	recommended that the coded character set be UTF-8 [UTF-8], the actual coded character set
1048 1049	SHALL be indicated by the value of the jobCodedCharSet (7) attribute for the job.
1049	NOTE - The values of objects and attributes using this textual convention are either generated
1050	by the job submitter or defaulted by the server or device when the job submitter does not supply
1052	values."
1053	REFERENCE
1054	"See section 3.5.2, "JmJobStringTC' for text generated by the job submitter'."

Aug 8, 1997

JmTimeStampTC ::= TEXTUAL-CONVENTION
 STATUS current DESCRIPTION "The simple time at which an event took place. The units SHALL be in seconds since the system was booted. NOTE - JmTimeStampTC is defined in units of seconds, rather than 100ths of seconds, so as to be simpler for agents to implement (even if they have to implement the 100ths of a second to comply with implementing sysUpTime in MIB-II[mib-II].) NOTE - JmTimeStampTC is defined as an Integer32 so that it can be used as a value of an attribute, i.e., as a value of the jmAttributeValueAsInteger object. The TimeStamp textual-convention defined in SMNPv2-TC is defined as an APPLICATION 3 IMPLICIT INTEGER tag, not an Integer32, so cannot be used in this MIB as one of the values of jmAttributeValueAsInteger." SYNTAX INTEGER(02147483647)
JmJobSourcePlatformTypeTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "The source platform type that can submit jobs to servers or devices in any of the 3 configurations." REFERENCE "This is a type 2 enumeration. See Section 3.6.1.2." SYNTAX INTEGER { other(1), unknown(2), sptUNIX(3), UNIX(tm) sptOS2(4), OS/2 sptPCDOS(5), DOS sptNT(6), NT sptMVS(7), MVS sptVM(8), VM sptOS400(9), OS/400 sptVM(8), VMS sptVMI(0), VMS sptVMI(1), Windows95 sptNetWare(33) NetWare }
J

Aug 8, 1997

1090	
1091	
1092	
1093	
1094	JmFinishingTC ::= TEXTUAL-CONVENTION
1095	STATUS current
1096	DESCRIPTION
1097	"The type of finishing operation.
1098	
1099	These values are the same as the enum values of the IPP 'finishings' attribute. See Section
1100	3.6.1.2.
1101	
1102	other(1),
1103	Some other finishing operation besides one of the specified or registered values.
1104	some suier missing sperator sestees one of the spectree of registered values.
1105	unknown(2),
1106	The finishing is unknown.
1100	The finishing is unknown.
1107	none(3),
1100	Perform no finishing.
1110	
1111	staple(4),
1112	Bind the document(s) with one or more staples. The exact number and placement of the
1112	staples is site-defined.
1113	suples is site defined.
1115	stapleTopLeft(5),
1115	Place one or more staples on the top left corner of the document(s).
1117	The one of more stuples on the top fert corner of the document(s).
1118	stapleBottomLeft(6),
1119	Place one or more staples on the bottom left corner of the document(s).
1120	There one of more supres on the bottom fert corner of the document(s).
1120	stapleTopRight(7),
1121	Place one or more staples on the top right corner of the document(s).
1122	Thee one of more suples on the top fight corner of the document(s).
1123	stapleBottomRight(8),
1124	Place one or more staples on the bottom right corner of the document(s).
1125	The one of more staples on the bottom right corner of the document(s).
1120	saddleStitch(9),
1127	Bind the document(s) with one or more staples (wire stitches) along the middle fold. The
1120	exact number and placement of the stitches is site-defined.
1129	exact number and placement of the stitches is site-defined.
1130	edgeStitch(10),
1131	Bind the document(s) with one or more staples (wire stitches) along one edge. The exact
1132	
1135	number and placement of the staples is site-defined.
1134	punch(11),
1135	This value indicates that holes are required in the finished document. The exact number
1130	and placement of the holes is site-defined The punch specification MAY be satisfied (in a
1137	and pracement of the noises is site-defined. The puttern specification what the satisfied (in a

1141 cover(12), 1142 This value is specified when it is desired to select a non-printed (or pre-printed) cover for the document. This does not supplant the specification of a printed cover (on cover stock medium) by the document itself. 1143 This value indicates that a binding is to be applied to the document; the type and placement of the binding is product-specific." 1144 This value indicates that a binding is product-specific." 1145 bind(13) 1146 Diacement of the binding is product-specific." 1147 This value indicates that a binding is robust specific." 1148 placement of the binding is product-specific." 1150 "This is a type 2 enumeration. See Section 3.6.1.2." 1151 stapleTopLight(7), stapleBottomLeft(6), stapleTopLight(7), stapleBottomLeft(6), stapleTopLight(7), stapleBottomLeft(6), stadleStitch(9), stapleTopLight(7), stapleBottomLeft(6), stadleStitch(9), stapleBottomLeft(6), stadleStitch(9), stapleBottomLeft(6), stadleStitch(9), stapleTopLight(7), stapleBottomLeft(6), stadleStitch(9), stapleBottomLeft(6), stadleStitch(9), stapleTopLight(7), stapleBottomLeft(6), stapleTopLight(7), stapleBottomLeft(1138 1139	site- and implementation-specific manner) either by drilling/punching, or by substituting pre-drilled media.
1141 cover(12), 1142 This value is specified when it is desired to select a non-printed (or pre-printed) cover for 1143 the document. This does not supplant the specification of a printed cover (on cover stock 1144 medium) by the document itself. 1145 bind(13) 1146 bind(13) 1147 This value indicates that a binding is to be applied to the document; the type and placement of the binding is product-specific." 1148 rThis is a type 2 enumeration. See Section 3.6.1.2." 1150 "This is a type 2 enumeration. See Section 3.6.1.2." 1151 SYNTAX INTEGER { 1152 other(1), 1153 unknown(2), 1154 none(3), 1155 stapleTopRight(7), 1158 stapleTopRight(7), 1159 stapleTopRight(7), 1160 edgeStitch(10), 1161 edgeStitch(10), 1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 j 1170 This value is a type 2 enumeration. See Section 3.6.1.2." 1171 DESCRIPTION <t< td=""><td></td><td></td></t<>		
1142 This value is specified when it is desired to select a non-printed (or pre-printed) cover for 1143 the document. This does not supplant the specification of a printed cover (on cover stock 1144 medium) by the document itself. 1145 bind(13) 1146 bind(13) 1147 This value indicates that a binding is to be applied to the document; the type and placement of the binding is product-specific." 1180 REFERENCE "This is a type 2 enumeration. See Section 3.6.1.2." 1151 SYNTAX INTEGER { 1152 other(1), 1153 unknown(2), 1154 none(3), 1155 staple(4), 1156 staple(4), 1157 stapleBottomLeft(6), 1158 staple(10), 1160 sadue/Estuch(9), 1161 edgeStitch(10), 1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 } 1166 ind(13) 1167 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 3.6.1.2." 1173 DESCRIPTIO		cover(12).
1143 the document, This does not supplant the specification of a printed cover (on cover stock medium) by the document itself. 1144 medium) by the document itself. 1145 bind(13) 1146 bind(13) 1147 This value indicates that a binding is to be applied to the document; the type and placement of the binding is product-specific." 1148 REFERENCE 1150 "This is a type 2 enumeration. See Section 3.6.1.2." 1151 SYNTAX 1152 other(1), 1153 unknown(2), 1154 none(3), 1155 stapleTopLeft(5), 1156 stapleTopLeft(6), 1158 stapleBottomRight(8), 1160 saddleStitch(9), 1161 edgestitch(10), 1162 panch(11), 1173 DESCRIPTION 1174 STATUS 1175 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 JBECRIPTION 1178 REFERENCE "This is a type 2 enumeration. See Se		
1144 medium) by the document itself. 1146 bind(13) 1147 This value indicates that a binding is to be applied to the document; the type and placement of the binding is product-specific." 1148 REFERENCE 1150 "This is a type 2 enumeration. See Section 3.6.1.2." 1151 SYNTAX INTEGER { 1152 other(1), 1153 unknown(2), 1154 none(3), 1155 stapleTopRight(7), 1158 stapleTopRight(7), 1159 stapleTopRight(7), 1160 adgeStitch(10), 1161 edgeStitch(10), 1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 } 1166 13 1170 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS 1173 DESCRIPTION 1174 "Print quality settings. 1175 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 3.6.1.2." 1174 REFERENCE		
1145 bind(13) 1147 This value indicates that a binding is to be applied to the document; the type and placement of the binding is product-specific." 1148 REFERENCE 1150 "This is a type 2 enumeration. See Section 3.6.1.2." 1151 SYNTAX 1152 other(1), 1153 unknown(2), 1154 none(3), 1155 staple(4), 1156 staple(10), 1157 stapleBottomLeft(6), 1158 stapleBottomRight(8), 1160 saddleStitch(9), 1161 edgeStitch(10), 1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 } 1170 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section		
1146 bind(13) 1147 This value indicates that a binding is to be applied to the document; the type and placement of the binding is product-specific." 1148 REFERENCE 1150 "This is a type 2 enumeration. See Section 3.6.1.2." 1151 SYNTAX 1152 other(1), 1153 unknown(2), 1154 none(3), 1155 staple(4), 1156 stapleTopRight(7), 1157 stapleTopRight(7), 1158 stapleTopRight(7), 1159 stapleTopRight(7), 1159 stapleTopRight(7), 1160 saddeStüch(9), 1161 edgeStüch(10), 1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 } 1170 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1171 JmPrintQuality settings. 1172 STATUS 1173 DESCRIPTION 1174 "Print quality settings. 1175 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section <t< td=""><td></td><td>medium) by the document risen.</td></t<>		medium) by the document risen.
1147This value indicates that a binding is to be applied to the document; the type and placement of the binding is product-specific."1149REFERENCE "This is a type 2 enumeration. See Section 3.6.1.2."1150"This is a type 2 enumeration. See Section 3.6.1.2."1151SYNTAX INTEGER { other(1),1152other(1),1153unknown(2),1154none(3),1155staple(4),1156staple(TopLeft(5),1157stapleBottomLeft(6),1158stapleFopRight(7),1159stapleBottomRight(8),1160saddleStitch(9),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165}1170JmPrintQualityTC ::= TEXTUAL-CONVENTION1171STATUS1172STATUS1173DESCRIPTION1174"Print quality settings.1175These values are the same as the enum values of the IPP 'print-quality' attribute. See Section11773.6.1.2."1178REFERENCE1179"This is a type 2 enumeration. See Section 3.6.1.2."1179"This is a type 2 enumeration. See Section 3.6.1.2."1179"This is a type 2 enumeration. See Section 3.6.1.2."1178REFERENCE1179"This is a type 2 enumeration. See Section 3.6.1.2."1180SYNTAX1171- Not one of the specified or registered values.1172- The actual value is unknown.1173undexeq		hind(13)
1148 placement of the binding is product-specific. ^{1,4} 1149 REFERENCE 1150 "This is a type 2 enumeration. See Section 3.6.1.2." 1151 SYNTAX 1152 other(1), 1153 unknown(2), 1154 none(3), 1155 staple(4), 1156 stapleTopRight(7), 1157 stapleTopRight(7), 1158 stapleTopRight(7), 1159 stapleTopRight(8), 1161 edgeStitch(10), 1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 } 1170 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1179 <td></td> <td></td>		
1149REFERENCE C T1150"This is a type 2 enumeration. See Section 3.6.1.2."1151SYNTAX INTEGER {1152other(1),1153unknown(2),1154none(3),1155stapleTopLeft(5),1157stapleBottomLeft(6),1158stapleTopLeft(5),1159stapleBottomRight(8),1160saddleStitch(9),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165}1170JmPrintQualityTC ::= TEXTUAL-CONVENTION1171JmPrintQualityTC surrent1172STATUS1173DESCRIPTION1174"Print quality settings.1175These values are the same as the enum values of the IPP 'print-quality' attribute. See Section1177 $3.6.1.2."$ 1178REFERENCE"This is a type 2 enumeration. See Section $3.6.1.2."$ 1180SYNTAX INTEGER {other(1), Not one of the specified or registered values		
1150"This is a type 2 enumeration. See Section 3.6.1.2."1151SYNTAX1152other(1),1153unknown(2),1154none(3),1155staple(4),1156stapleTopLeft(5),1157stapleBottomRight(8),1160saddleStitch(10),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165}1170JmPrintQualityTC ::= TEXTUAL-CONVENTION1171STATUS1172STATUS1173DESCRIPTION1174"Print quality settings.1175These values are the same as the enum values of the IPP 'print-quality' attribute. See Section1177 $3.6.1.2.$ "1180SYNTAXSYNTAXINTEGER {other(1),Not one of the specified or registered values <td></td> <td></td>		
1151SYNTAXINTEGER { other(1),1152other(1),1153unknown(2),1154none(3),1155staple(4),1156stapleTopLeft(5),1157stapleBottomLeft(6),1158stapleBottomLeft(6),1159stapleBottomRight(8),1160saddleStitch(9),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165}116611671170JmPrintQualityTC ::= TEXTUAL-CONVENTION1171JmPrintQualityTC ::= TEXTUAL-CONVENTION1172STATUScurrent1173DESCRIPTION1174"Print quality settings.11751176These values are the same as the enum values of the IPP 'print-quality' attribute. See Section11771178REFERENCE1179"This is a type 2 enumeration. See Section 3.6.1.2."1180SYNTAXSYNTAXINTEGER { other(1), Not one of the specified or registered values <td< td=""><td></td><td></td></td<>		
1152other(1),1153unknown(2),1154none(3),1155stapleTopLett(5),1156stapleTopRight(7),1159stapleBottomLeft(6),1158stapleBottomLeft(6),1159stapleBottomRight(8),1160sadeleStitch(10),1161edgeStitch(10),1162punch(11),1163covert(12),1164bind(13)1165}1166116711701171JmPrintQualityTC ::= TEXTUAL-CONVENTION1172STATUScurrent1173DESCRIPTION1174"Print quality settings.11751176These values are the same as the enum values of the IPP 'print-quality' attribute. See Section11773.6.1.2."1180SYNTAXSYNTAXINTEGER {other(1),unknown(2),unknown(2),unknown(2),unknown(2),unknown(2), </td <td></td> <td></td>		
1153unknown(2),1154none(3),1155staple(4),1156staple(10),1157stapleBottomLeft(6),1158stapleBottomRight(7),1159stapleBottomRight(8),1160saddleStitch(9),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165}116611671168116911701171JmPrintQualityTC ::= TEXTUAL-CONVENTION1172STATUScurrent1173DESCRIPTION1174"Print quality settings.11751176These values are the same as the enum values of the IPP 'print-quality' attribute. See Section11773.6.1.2."1178REFERENCE1179"This is a type 2 enumeration. See Section 3.6.1.2."1180SYNTAXSYNTAX INTEGER {other(1),nknown(2),unknown(2), <td< td=""><td></td><td></td></td<>		
1154none(3),1155staple(4),1156stapleTopLeft(5),1157stapleBottomLeft(6),1158stapleTopRight(7),1159stapleBottomRight(8),1160saddleStitch(10),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165}116616611671168116811691170STATUS1171JmPrintQualityTC ::= TEXTUAL-CONVENTION1172STATUSSTATUScurrent1173DESCRIPTION1174"Print quality settings.1175These values are the same as the enum values of the IPP 'print-quality' attribute. See Section1176These values are the same as the enum values of the IPP 'print-quality' attribute. See Section11773.6.1.2."1178REFERENCE1179"This is a type 2 enumeration. See Section 3.6.1.2."1180SYNTAXSYNTAXINTEGER {other(1),The actual value is unknown		
1155staple(4),1156stapleTopLeft(5),1157stapleBottomLeft(6),1158stapleBottomRight(8),1160sadeleStitch(9),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165}11661661167These values are the same as the enum values of the IPP 'print-quality' attribute. See Section1171JmPrint quality settings.1173DESCRIPTION1174"Print quality settings.1175These values are the same as the enum values of the IPP 'print-quality' attribute. See Section1178REFERENCE1179"This is a type 2 enumeration. See Section 3.6.1.2."1180SYNTAXSYNTAXINTEGER {other(1),other(1),The actual value is unknown.draft(3),Lowest quality available on the printer.		
1156stapleTopLeft(5),1157stapleBottomLeft(6),1158stapleTopRight(7),1159stapleBottomRight(8),1160saddleStitch(9),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165}1166116711681168116911701171JmPrintQualityTC ::= TEXTUAL-CONVENTION1172STATUS current1173DESCRIPTION1174"Print quality settings.11751176These values are the same as the enum values of the IPP 'print-quality' attribute. See Section11773.6.1.2."1178REFERENCE1179"This is a type 2 enumeration. See Section 3.6.1.2."1180SYNTAX INTEGER {other(1), Not one of the specified or registered values		
1157stapleBottomLeft(6),1158stapleTopRight(7),1159stapleBottomRight(8),1160saddleStitch(9),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165 $\}$ 116611671168116911701171JmPrintQualityTC ::= TEXTUAL-CONVENTION1172STATUScurrent1173DESCRIPTION1174"Print quality settings.11751176These values are the same as the enum values of the IPP 'print-quality' attribute. See Section11773.6.1.2."1178REFERENCE1179"This is a type 2 enumeration. See Section 3.6.1.2."1180SYNTAXSYNTAXINTEGER {other(1),<		
1158stapleTopRight(7),1159stapleBottomRight(8),1160saddleStitch(9),1161edgeStitch(10),1162punch(11),1163cover(12),1164bind(13)1165 $\}$ 1166116711701171JmPrintQualityTC ::= TEXTUAL-CONVENTION1172STATUS2017current1173DESCRIPTION1174"Print quality settings.1175These values are the same as the enum values of the IPP 'print-quality' attribute. See Section1176These values are the same as the enum values of the IPP 'print-quality' attribute. See Section1176These values are the same as the enum values of the IPP 'print-quality' attribute. See Section1175These values are the same as the enum values of the IPP 'print-quality' attribute. See Section1174"Print quality settings.1175These values are the same as the enum values of the IPP 'print-quality' attribute. See Section11761180SYNTAXINTEGER {other(1),Not one of the specified or registered values <td></td> <td></td>		
1159 stapleBottomRight(8), 1160 saddleStitch(9), 1161 edgeStitch(10), 1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 } 1166 1 1167 Iffee 1168 1169 1170 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1171 JmStatus 1172 STATUS 1173 DESCRIPTION 1174 "Print quality settings. 1175 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1),		
1160 saddleStitch(9), 1161 edgeStitch(10), 1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 } 1166		
1161 edgeStitch(10), 1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 } 1166		
1162 punch(11), 1163 cover(12), 1164 bind(13) 1165 } 1166 1 1167 1 1168 1 1169 1 1170 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1171 JmSCRIPTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1175 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Intervent(2), The actual value is unknown. unknown(2), Lowest quality available on the		
<pre>1163 cover(12), 1164 bind(13) 1165 } 1166 1167 1168 1169 1170 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values.</pre>		
1164 bind(13) 1165 } 1166		
<pre>1165 } 1166 1167 1168 1169 1170 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values.</pre>		
1166 1167 1168 1169 1170 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { 1179 other(1), Not one of the specified or registered values. 1170 Not one of the specified or registered values. 1171		
1167 1168 1169 1170 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { 0 other(1), Not one of the specified or registered values		J
1168 1169 1170 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { 0 ther(1), Not one of the specified or registered values. unknown(2), The actual value is unknown. draft(3), Lowest quality available on the printer.		
1169 1170 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { 0ther(1), Not one of the specified or registered values. unknown(2), The actual value is unknown. draft(3), Lowest quality available on the printer.		
1170 1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { 0ther(1), Not one of the specified or registered values. unknown(2), The actual value is unknown. draft(3), Lowest quality available on the printer.		
1171 JmPrintQualityTC ::= TEXTUAL-CONVENTION 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { 0ther(1), Not one of the specified or registered values. unknown(2), The actual value is unknown. draft(3), Lowest quality available on the printer.		
 1172 STATUS current 1173 DESCRIPTION 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values. unknown(2), The actual value is unknown. draft(3), Lowest quality available on the printer. 	1170	
1173 DESCRIPTION 1174 "Print quality settings. 1175 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values. unknown(2), Lowest quality available on the printer.	1171	JmPrintQualityTC ::= TEXTUAL-CONVENTION
 1174 "Print quality settings. 1175 1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values. The actual value is unknown. draft(3), Lowest quality available on the printer. 	1172	STATUS current
 1175 1176 1176 1176 1178 1178 1178 1179 1179 1180 1170 1180 1170 1180 1180<td>1173</td><td>DESCRIPTION</td>	1173	DESCRIPTION
1176 These values are the same as the enum values of the IPP 'print-quality' attribute. See Section 3.6.1.2." 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values. unknown(2), The actual value is unknown. draft(3), Lowest quality available on the printer.	1174	"Print quality settings.
 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values.	1175	
 1177 3.6.1.2." 1178 REFERENCE 1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values.	1176	These values are the same as the enum values of the IPP 'print-quality' attribute. See Section
1179 "This is a type 2 enumeration. See Section 3.6.1.2." 1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values. unknown(2), The actual value is unknown. draft(3), Lowest quality available on the printer.	1177	
1180 SYNTAX INTEGER { other(1), Not one of the specified or registered values. unknown(2), The actual value is unknown. draft(3), Lowest quality available on the printer.	1178	
other(1),Not one of the specified or registered values.unknown(2),The actual value is unknown.draft(3),Lowest quality available on the printer.	1179	"This is a type 2 enumeration. See Section 3.6.1.2."
unknown(2), The actual value is unknown. draft(3), Lowest quality available on the printer.	1180	
unknown(2), draft(3),The actual value is unknown. Lowest quality available on the printer.		other(1), Not one of the specified or registered values.
draft(3), Lowest quality available on the printer.		
normal(4), Normal or intermediate quality on the printer.		
		normal(4), Normal or intermediate quality on the printer.

--

	high(5) Highest quality available on the printer.
1181 1182 1183 1184	}
1185	
1186 1187 1188 1189 1190 1191 1192 1193 1194 1195 1196 1197 1198 1199 1200 1201 1202 1203 1204 1205	<pre>JmPrinterResolutionTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "Printer resolutions. Nine octets consisting of two 4-octet SIGNED-INTEGERs followed by a SIGNED-BYTE. The values are the same as those specified in the Printer MIB [printmib]. The first SIGNED-INTEGER contains the value of prtMarkerAddressabilityXFeedDir. The second SIGNED-INTEGER contains the value of prtMarkerAddressabilityFeedDir. The SIGNED-BYTE contains the value of prtMarkerAddressabilityFeedDir. The SIGNED-BYTE contains the value of prtMarkerAddressabilityFeedDir. The SIGNED-BYTE contains the value of prtMarkerAddressabilityUnit. Note: the latter value is either 3 (tenThousandsOfInches) or 4 (micrometers) and the addressability is in 10,000 units of measure. Thus the SIGNED-INTEGERs represent integral values in either dots-per-inch or dots-per-centimeter. The syntax is the same as the IPP 'printer-resolution' attribute. See Section 3.6.1.2." SYNTAX OCTET STRING (SIZE(9))</pre>
1206	
1207 1208 1209 1210 1211 1212 1213 1214	JmTonerEconomyTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "Toner economy settings." REFERENCE "This is a type 2 enumeration. See Section 3.6.1.2." SYNTAX INTEGER { unknown(2), unknown. off(3), Off. Normal. Use full toner. on(4) On. Use less toner than normal.
1215 1216 1217 1218 1219 1220 1221 1222	} JmBooleanTC ::= TEXTUAL-CONVENTION
1222	STATUS current

1224 "Boolean true or false value." 1225 REFERENCE "This is a type 2 enumeration. See Section 3.6.1.2." 1227 SYNTAX INTEGER {	1223	DESCRIPTION
1225 REFERENCE 1226 "This is a type 2 enumeration. See Section 3.6.1.2." 1227 SYNTAX_INTEGER { unknown(2), - unknown false(3), FALSE. true(4) - TRUE. 1231 ImMediumTypeTC ::= TEXTUAL-CONVENTION 1233 STATUS current 1234 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1242 unknown(2), 1243 The type is not known. 1244 Separately cut sheets of an opaque material. 1247 Transparency(4), 1248 transparency(4), 1249 Separately cut sheets of a transparent material. 1250 Envelopes that can be used for conventional mailing purposes. 1251 Envelopes that are not preprinted and have no windows. 1255		
1226 "This is a type 2 enumeration. See Section 3.6.1.2." 1227 SYNTAX INTEGER { <unknown. <="" false(3),="" false.="" td="" true(4)="" true.=""> 1228 } 1229 1230 1231 1232 1233 1234 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS 1236 1237 "Identifies the type of medium. 1238 1239 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 1242 unknown(2), 1244 1245 stationery(3), 1246 Separately cut sheets of a transparent material. 1250 Envelopes that can be used for conventional mailing purposes. 1251 envelopes (6), 1252 Envelopes that are not preprinted and have no windows. 1253 Envelopes that are not preprinted and have no windows. 1257 <td></td><td></td></unknown.>		
1227 SYNTAX INÎÊGER {		
unknown(2), unknown. false(3), FALSE. 1229 - TRUE. 1230 TRUE. 1231 TRUE. 1232 TRUE. 1233 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 Unknown(2), 1244 Separately cut sheets of an opaque material. 1244 Separately cut sheets of a transparent material. 1245 separately cut sheets of a transparent material. 1250 Envelope(5), 1251 envelope(5), 1252 Envelopes that can be used for conventional mailing purposes. 1253 Envelope(5), 1254 envelope(7), 1255 Envelopes that are not preprinted and have no windows. 1256 Envelopes that have windows for addressing purposes. 1257 envelope(8),		
false(3), FALSE. 1228 - 1229 - 1230 - 1231 - 1232 - 1233 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 'Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241		
true(4) TRUE. 1229 1 1230 1 1231 1 1232 1 1233 1 1234 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1243 The type is not known. 1244 Separately cut sheets of an opaque material. 1245 stationery(3), 1246 Separately cut sheets of a transparent material. 1251 envelope(5), 1252 Envelopes that can be used for conventional mailing purposes. 1253 Envelope(5), 1254 envelope(5), 1255 Envelopes that are not preprinted and have no windows. 1256 Envelopes that are not preprinted and have no windows. 1256 Envelope(8), 1257 Envelope(8), 1258 Envelope(8),		
1228 } 1230 1231 1233 1232 1234 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1242 unknown(2), 1243 The type is not known. 1244 Separately cut sheets of an opaque material. 1245 stationery(3), 1246 Separately cut sheets of a transparent material. 1250 Envelope(5), 1251 envelope(5), 1252 Envelopes that can be used for conventional mailing purposes. 1253 Envelope(6), 1254 envelopePlain(6), 1255 Envelopes that are not preprinted and have no windows. 1256 Envelopes (7), 1258 Envelope (7), 1258 Envelopes (7), 1259 Envelopes (7), 1258 Envelopes (7), <		
1229 . 1231 131 1232 1233 1233 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1239 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1243 The type is not known. 1244 Separately cut sheets of an opaque material. 1244 Separately cut sheets of a transparent material. 1249 Separately cut sheets of a transparent material. 1250 Envelopes that can be used for conventional mailing purposes. 1251 envelope[6], 1255 Envelopes that are not preprinted and have no windows. 1256 Envelopes that are not preprinted and have no windows. 1256 envelope[1] 1257 envelope[2] 1258 Envelopes that have windows for addressing purposes. 1259 ContinuousUng(8), 1259 ContinuousUs connected sheets of an opaque material connected along the long	1228	
1230 1231 1232 1233 1234 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 1239 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 1242 unknown(2), 1243 The type is not known. 1244 1245 stationery(3), 1246 Separately cut sheets of an opaque material. 1250 Envelope(5), Envelopes that can be used for conventional mailing purposes. 1251 envelopePlain(6), EnvelopeWindow(7), EnvelopeWindow(7), EnvelopeWindow(7), EnvelopeShat hare windows for addressing purposes. 1259 1250 1251 1252 1253 1254 <td></td> <td>J</td>		J
1231 1233 1234 JmMediumTypeTC := TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1242 unknown(2), 1243 The type is not known. 1244 separately cut sheets of an opaque material. 1246 Separately cut sheets of a transparent material. 1250 envelope(5), 1251 envelopes that can be used for conventional mailing purposes. 1252 Envelopes that are not preprinted and have no windows. 1255 Envelopes that are not preprinted and have no windows. 1256 envelopeWindow(7), 1258 Envelopes that have windows for addressing purposes. 1259 continuousLong(8), 1261 ContinuousShort(9), 1262 continuously connected sheets of an opaque material connected along the short edge.		
1232 1233 1234 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1239 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1242 transparency(3), 1244 Separately cut sheets of an opaque material. 1247 Separately cut sheets of a transparent material. 1248 transparency(4), 1250 Envelope(5), 1251 envelope(5), 1252 Envelopes that can be used for conventional mailing purposes. 1253 Envelopes that are not preprinted and have no windows. 1256 Envelopes that have windows for addressing purposes. 1257 Envelopes that have windows for addressing purposes. 1259 Envelopes that have windows for addressing purposes. 1259 Envelopes that have windows for addressing purposes. 1259 ContinuousLong(8), 1260 ContinuousShort(9), <tr< td=""><td></td><td></td></tr<>		
1233 1234 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1242 unknown(2), 1243 The type is not known. 1244 Separately cut sheets of an opaque material. 1245 stationery(3), 1246 Separately cut sheets of a transparent material. 1247 Envelope(5), 1250 Envelopes that can be used for conventional mailing purposes. 1253 Envelopes that are not preprinted and have no windows. 1254 Envelopes that are not preprinted and have no windows. 1255 Envelopes that have windows for addressing purposes. 1256 Envelopes that have windows for addressing purposes. 1257 Envelopes that have windows for addressing purposes. 1256 Envelopes that have windows for addressing purposes. 1257 Envelopes that have windows for addressing purposes. 1258 Envelopes that hav		
1234 JmMediumTypeTC ::= TEXTUAL-CONVENTION 1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1242 unknown(2), 1244 Separately cut sheets of an opaque material. 1245 stationery(3), 1246 Separately cut sheets of a transparent material. 1250 envelope(5), 1251 envelope(5), 1252 Envelopes that can be used for conventional mailing purposes. 1255 Envelopes that are not preprinted and have no windows. 1256 envelopeWindow(7), 1258 Envelopes that have windows for addressing purposes. 1259 continuously connected sheets of an opaque material connected along the long edge. 1260 continuously connected sheets of an opaque material connected along the short edge.		
1235 STATUS current 1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1242 unknown(2), 1243 The type is not known. 1244 Separately cut sheets of an opaque material. 1245 stationery(3), 1246 Separately cut sheets of a transparent material. 1250 envelope(5), 1251 envelope(5), 1252 Envelopes that can be used for conventional mailing purposes. 1255 Envelopes that are not preprinted and have no windows. 1256 envelopeWindow(7), 1258 Envelopes that have windows for addressing purposes. 1259 Continuously connected sheets of an opaque material connected along the long edge. 1260 continuously connected sheets of an opaque material connected along the short edge. 1261 Continuously connected sheets of an opaque material connected along the short edge.	1233	
1236 DESCRIPTION 1237 "Identifies the type of medium. 1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1243 The type is not known. 1244 stationery(3), 1245 stationery(4), 1246 Separately cut sheets of a ransparent material. 1247 transparency(4), 1248 transparency(4), 1250 Envelope(5), 1251 envelope(5), 1253 Envelopes that can be used for conventional mailing purposes. 1255 Envelopes that are not preprinted and have no windows. 1257 envelopeWindow(7), 1258 Envelopes that have windows for addressing purposes. 1259 ContinuousLong(8), 1261 Continuously connected sheets of an opaque material connected along the long edge. 1261 continuously connected sheets of an opaque material connected along the short edge. 1262 continuously connected sheets of an opaque material connected along the short edge.	1234	JmMediumTypeTC ::= TEXTUAL-CONVENTION
1237"Identifies the type of medium.1238other(1),1240The type is neither one of the values listed in this specification nor a registered value.1241unknown(2),1242unknown(2),1243The type is not known.1244stationery(3),1245stationery(3),1246Separately cut sheets of an opaque material.1247Separately cut sheets of a transparent material.1248transparency(4),1249Separately cut sheets of a transparent material.1250Envelope(5),1251envelope(5),1252Envelopes that can be used for conventional mailing purposes.1253envelopePlain(6),1254envelopePlain(6),1255Envelopes that are not preprinted and have no windows.1256envelopeWindow(7),1258Envelopes that have windows for addressing purposes.1259ContinuousLong(8),1261ContinuousShort(9),1262ContinuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.	1235	
1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1243 The type is not known. 1244 stationery(3), 1246 Separately cut sheets of an opaque material. 1247 ransparency(4), 1248 transparency(4), 1250 envelope(5), 1251 envelopes that can be used for conventional mailing purposes. 1253 Envelopes that are not preprinted and have no windows. 1257 envelopeWindow(7), 1258 Envelopes that have windows for addressing purposes. 1259 ContinuousLong(8), 1261 continuously connected sheets of an opaque material connected along the long edge. 1262 continuously connected sheets of an opaque material connected along the short edge.	1236	DESCRIPTION
1238 other(1), 1240 The type is neither one of the values listed in this specification nor a registered value. 1241 unknown(2), 1243 The type is not known. 1244 stationery(3), 1245 stationery(4), 1246 Separately cut sheets of a ransparent material. 1247 ransparency(4), 1248 transparency(4), 1250 Envelope(5), 1251 envelope(5), 1252 Envelopes that can be used for conventional mailing purposes. 1253 Envelopes that are not preprinted and have no windows. 1255 Envelopes that are not preprinted and have no windows. 1256 envelopeWindow(7), 1258 Envelopes that have windows for addressing purposes. 1259 ContinuousLong(8), 1261 Continuously connected sheets of an opaque material connected along the long edge. 1262 continuously connected sheets of an opaque material connected along the short edge.	1237	"Identifies the type of medium.
1240The type is neither one of the values listed in this specification nor a registered value.1241unknown(2),1243The type is not known.124412451245stationery(3),1246Separately cut sheets of an opaque material.1247Itransparency(4),1248transparency(4),1250Separately cut sheets of a transparent material.1250Envelope(5),1251envelope(5),1252Envelopes that can be used for conventional mailing purposes.1253Envelopes that are not preprinted and have no windows.1256Envelopes that have windows for addressing purposes.1259ContinuousLong(8),1261ContinuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.	1238	
1240The type is neither one of the values listed in this specification nor a registered value.1241unknown(2),1243The type is not known.124412451245stationery(3),1246Separately cut sheets of an opaque material.1247Transparency(4),1248transparency(4),1250Separately cut sheets of a transparent material.1250Envelope(5),1251envelope(5),1252Envelopes that can be used for conventional mailing purposes.1253Envelopes that are not preprinted and have no windows.1256Envelopes that have windows for addressing purposes.1259ContinuousLong(8),1261ContinuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.	1239	other(1),
1241unknown(2),1243The type is not known.1244The type is not known.1244stationery(3),1245stationery(3),1246Separately cut sheets of an opaque material.1247Transparency(4),1248transparency(4),1250Separately cut sheets of a transparent material.1250Envelope(5),1251envelope(5),1252Envelopes that can be used for conventional mailing purposes.1253Envelopes that are not preprinted and have no windows.1256Envelopes that are not preprinted and have no windows.1257envelopeWindow(7),1258Envelopes that have windows for addressing purposes.1259ContinuousLong(8),1260ContinuousShort(9),1261ContinuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.	1240	
1243The type is not known.124412451245stationery(3),1246Separately cut sheets of an opaque material.124712481248transparency(4),1249Separately cut sheets of a transparent material.1250envelope(5),1251envelope(5),1252Envelopes that can be used for conventional mailing purposes.1253envelopePlain(6),1255Envelopes that are not preprinted and have no windows.1256envelopeWindow(7),1258Envelopes that have windows for addressing purposes.1259ContinuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.1262continuously connected sheets of an opaque material connected along the short edge.1263continuously connected sheets of an opaque material connected along the short edge.		
1243The type is not known.124412451245stationery(3),1246Separately cut sheets of an opaque material.124712481248transparency(4),1249Separately cut sheets of a transparent material.1250envelope(5),1251envelope(5),1252Envelopes that can be used for conventional mailing purposes.1253envelopePlain(6),1255Envelopes that are not preprinted and have no windows.1256envelopeWindow(7),1258Envelopes that have windows for addressing purposes.1259ContinuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.1262continuously connected sheets of an opaque material connected along the short edge.1263continuously connected sheets of an opaque material connected along the short edge.	1242	unknown(2),
124412451245stationery(3),1246Separately cut sheets of an opaque material.124712481248transparency(4),1249Separately cut sheets of a transparent material.1250envelope(5),1251envelope(5),1252Envelopes that can be used for conventional mailing purposes.1253envelopePlain(6),1255Envelopes that are not preprinted and have no windows.1256envelopeWindow(7),1258Envelopes that have windows for addressing purposes.1259continuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.126212631264Continuously connected sheets of an opaque material connected along the short edge.		
1245stationery(3),1246Separately cut sheets of an opaque material.12471248transparency(4),1249Separately cut sheets of a transparent material.12501251envelope(5),1252Envelopes that can be used for conventional mailing purposes.12531254envelopePlain(6),1255Envelopes that are not preprinted and have no windows.12561257envelopeWindow(7),1258Envelopes that have windows for addressing purposes.1259ContinuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.1263continuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		
1246Separately cut sheets of an opaque material.124712481249124912501251envelope(5),12521254envelopePlain(6),125512561257envelopeWindow(7),125812591260continuousLong(8),126112621263continuousShort(9),12641264		stationerv(3).
12471248transparency(4),1249Separately cut sheets of a transparent material.125012511251envelope(5),1252Envelopes that can be used for conventional mailing purposes.125312541255EnvelopePlain(6),1256Envelopes that are not preprinted and have no windows.125612571257envelopeWindow(7),1258Envelopes that have windows for addressing purposes.125912601260continuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.1263continuously connected sheets of an opaque material connected along the short edge.		
1248transparency(4),1249Separately cut sheets of a transparent material.1250envelope(5),1251envelope(5),1252Envelopes that can be used for conventional mailing purposes.1253envelopePlain(6),1255Envelopes that are not preprinted and have no windows.1256envelopeWindow(7),1257envelopes that have windows for addressing purposes.1259ContinuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.1262ContinuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		r r r r r
1249Separately cut sheets of a transparent material.12501251125112521253125312541255125512561257125812581259126012611261126212621263126412651264126412651265126412641265126412641265126512661267126812681269126412641264126512651265126512651265126512651265126512651265126512661267126812691269126012601260<		transparency(4).
1250Interventional mailing purposes1251envelope(5),1252Envelopes that can be used for conventional mailing purposes.1253intervelopePlain(6),1255Envelopes that are not preprinted and have no windows.1256intervelopeWindow(7),1257envelopeWindow(7),1258Envelopes that have windows for addressing purposes.1259intervelope(8),1260continuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.1263continuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		
1251envelope(5), Envelopes that can be used for conventional mailing purposes.1252Envelopes that can be used for conventional mailing purposes.1253envelopePlain(6), Envelopes that are not preprinted and have no windows.1256envelopeWindow(7), Envelopes that have windows for addressing purposes.1259continuousLong(8), Continuously connected sheets of an opaque material connected along the long edge.1263continuousShort(9), Continuously connected sheets of an opaque material connected along the short edge.		
1252Énvelopes that can be used for conventional mailing purposes.1253125412551256125612571258125912601261126112621262126312641265126412641265126412641264126512651266126712681268126912691260126012611262126312641264126412651265126512651265126512651265126512661267126812681269126912601260126012601260126012		envelope(5).
1253Image: Continuous Continuo		
1254envelopePlain(6),1255Envelopes that are not preprinted and have no windows.125612571257envelopeWindow(7),1258Envelopes that have windows for addressing purposes.125912601260continuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.126212631264Continuously connected sheets of an opaque material connected along the short edge.		
1255Envelopes that are not preprinted and have no windows.12561257125712581259126012611262126212631264Continuously connected sheets of an opaque material connected along the short edge.		envelopePlain(6).
1256Image: Continuous Long(8),1257Continuous Long(8),1260Continuous long(8),1261Continuous long(8),1262Continuous long(9),1263Continuous Short(9),1264Continuous lo connected sheets of an opaque material connected along the short edge.		
1257envelopeWindow(7),1258Envelopes that have windows for addressing purposes.125912601260continuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.126212631263continuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		I I I I I I I I I I I I I I I I I I I
1258Énvelopes that have windows for addressing purposes.125912601260continuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.126212631263continuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		envelopeWindow(7).
1259continuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.1262continuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		
1260continuousLong(8),1261Continuously connected sheets of an opaque material connected along the long edge.1262continuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		
1261Continuously connected sheets of an opaque material connected along the long edge.1262continuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		continuousLong(8).
1262continuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		
1263continuousShort(9),1264Continuously connected sheets of an opaque material connected along the short edge.		Continuousi, connected sneets of an opaque material connected along the long cage.
1264 Continuously connected sheets of an opaque material connected along the short edge.		continuousShort(9).
1265	1265	Continuously connected sheets of an opaque material connected along the short edge.
1266 tabStock(10),		tabStock(10).
1267 Media with tabs.		

1268 1269 1270 1271 1272	multiPartForm(11), Form medium composed of multiple layers not pre-attached to one another; each sheet MAY be drawn separately from an input source.
1272 1273 1274 1275	labels(12), Label-stock.
1275 1276 1277 1278	multiLayer(13) Form medium composed of multiple layers which are pre-attached to one another, e.g. for use with impact printers."
1279 1280 1281	REFERENCE "This is a type 2 enumeration. See Section 3.6.1.2." SYNTAX INTEGER {
1281 1282 1283 1284	other(1), unknown(2), stationery(3),
1285 1286 1287	transparency(4), envelope(5), envelopePlain(6),
1288 1289 1290 1291 1292	envelopeWindow(7), continuousLong(8), continuousShort(9), tabStock(10), multiPartForm(11),
1293 1294 1295 1296	labels(12), multiLayer(13) }
1297 1298 1299	
1300 1301 1302 1303	JmJobSubmissionTypeTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION
1304 1305 1306	"Identifies the format type of a job submission ID. The ASCII characters '0-9', 'A-Z', and 'a-z' are assigned in order giving 62 possible formats.
1307 1308 1309 1310	Each job submission ID is a fixed-length, 48-octet printable ASCII coded character string, consisting of the following fields:
1310 1311 1312 1313 1314	octet 1 The format letter. octets 2-40 A 39-character, ASCII trailing SPACE filled field specified by the format letter, if the data is less than 39 ASCII characters.
1314	octets 41-48 A sequential or random number to make the ID

Aug 8, 1997

quasi-unique.

If the client does not supply a job submission ID in the job submission protocol, then the server SHALL assign a job submission ID using any of the standard formats that are reserved to the agent. Clients SHALL not use formats that are reserved to agents.

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

Format

1316

1317 1318

1319 1320

1321 1322

1323 1324

1325

Letter Description

1325	Letter	Description
1326		
1327	' 0'	octets 2-40: last 39 bytes of the jmJobOwner
1328		object.
1329		octets 41-48: 8-decimal-digit sequential number
1330		This format is reserved to agents for use when
1331		the client does not supply a job submission ID.
1332		Clients wishing to use a job submission ID that
1333		incorporates the job owner, SHALL use format '8',
1334		not format '0', in order to reduce the chances of
1335		one client assigning the same ID as the agent when
1336		receiving a job from another client that does not
1337		supply a job submission id.
1338		
1339		NOTE - other formats may be registered that are
1340		reserved to the agent for use when the client does
1341		not supply a job submission ID.
1342		
1343	'1'	octets 2-40: last 39 bytes of the jobName attribute.
1344		octets 41-48: 8-decimal-digit random number
1345		-
1346	'2'	octets 2-40: Client MAC address: in hexadecimal
1347		with each nibble of the 6 octet address being
1348		'0'-'9' or 'A' - 'F' (uppercase only).
1349		Most significant octet first.
1350		octets 41-48: 8-decimal-digit sequential number
1351		
1352	'3'	octets 2-40: last 39 bytes of the client URL
1353		[URI-spec].
1354		octets 41-48: 8-decimal-digit sequential number
1355		
1356	' 4'	octets 2-40: last 39 bytes of the URI [URI-spec]
1357		assigned by the server or device to the job when
1358		the job was submitted for processing.
1359		octets 41-48: 8-decimal-digit sequential number
1360		
1361	'5'	octets 2-40: last 39 bytes of a user number, such
1362		as POSIX user number.
1363		octets 41-48: 8-decimal-digit sequential number
1364		

1411	Figure 4 - Normal Job State Transitions
1410	
1409	+> pendingHeld(4) processingStopped(6)+
1408	v v /
1407	>+ +> aborted(8)
1406	
1405	+> pending(3)> processing(5)> completed(9)
1404	/
1403	+> canceled(7)
1402	
1401	The following figure shows the normal job state transitions:
1400	
1399	"The current state of the job (pending, processing, completed, etc.).
1398	DESCRIPTION
1397	STATUS current
1396	JmJobStateTC ::= TEXTUAL-CONVENTION
1395	
1394	
1393	
1392	
1391	
1390	SYNTAX OCTET STRING(SIZE(1)) ASCII '0'-'9', 'A'-'Z', 'a'-'z'
1389	"This is like a type 2 enumeration. See section 3.6.3."
1388	REFERENCE
1387	monitoring jobs, not for controlling and managing them."
1385	possible, but without bad consequences, since this MIB is intended to be used only for
1385	intended to be an absolute guarantee of uniqueness. None-the-less, collisions are remotely
1385	been selected to reduce the probability of collision to an extremely low number, but is not
1382 1383	such as the MAC address or URL, a sequential number SHOULD suffice for each client (and may be easier for each client to manage). Therefore, the length of the job submission id has
1381	submission id. For other formats, where part of the id is guaranteed to be unique for each client,
1380	random number so that the same job submitted by the same client will have a different job
1379	that is processing the job. Some of the formats include something that is unique per client and a
1378	limited duration of time, namely, for the life time of the job in the context of the server or device
1377	NOTE - the job submission id is only intended to be unique between a limited set of clients for a
1376	
1375	octets 41-48: 8-decimal-digit sequential number
1374	(that the agent returns in the jmJobOwner object).
1372	'8' octets 2-40: last 39 bytes of the job owner name
1371	ocicis +1-+0. o-ucciliai-uigit sequentiai number
1370	octets 41-48: 8-decimal-digit sequential number
1369 1370	'7' octets 2-40: last 39 bytes of the DTMF incoming FAX routing number.
1368	'7' octets 2-40: last 39 bytes of the DTMF incoming
1367	octets 41-48: 8-decimal-digit sequential number
1366	number.
1365	'6' octets 2-40: last 39 bytes of the user account

1412 1413 Normally a job progresses from left to right. Other state transitions are unlikely, but are not 1414 forbidden. Not shown are the transitions to the **canceled** state from the **pending**. 1415 pendingHeld, processing, and processingStopped states. 1416 1417 Jobs in the **pending**, **processing**, and **processingStopped** states are called 'active', while jobs in the pendingHeld, canceled, aborted, and completed are called 'inactive'. 1418 1419 1420 These values are the same as the enum values of the IPP 'job-state' job attribute. See Section 1421 3.6.1.2. 1422 1423 unknown(2). 1424 The job state is *not* known, or its state is indeterminate. 1425 1426 pending(3), 1427 The job is a candidate to start processing, but is not yet processing. 1428 1429 pendingHeld(4), 1430 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 1431 1432 **jmJobStateReasons1** object and/or **jobStateReasons** (N=2..4) attributes SHALL 1433 indicate why the job is no longer a candidate for processing. The reasons are represented 1434 as bits in the **jmJobStateReasons1** object and/or jobStateReasonsN (N=2..4) attributes. 1435 See the **JmJobStateReasonsNTC** (*N*=1..4) textual convention for the specification of 1436 each reason. 1437 1438 processing(5), 1439 Either: 1440 1441 1. The job is using, or is attempting to use, one or more document transforms which 1442 include (1) purely software processes that are interpreting a PDL, and (2) hardware 1443 devices that are interpreting a PDL, making marks on a medium, and/or performing 1444 finishing, such as stapling, etc. 1445 1446 OR 1447 2. (configuration 2) the server has made the job ready for printing, but the output device is 1448 1449 not yet printing it, either because the job hasn't reached the output device or because the 1450 job is queued in the output device or some other spooler, awaiting the output device to 1451 print it. 1452 1453 When the job is in the **processing** state, the entire job state includes the detailed status 1454 represented in the device MIB indicated by the **hrDeviceIndex** value of the job's 1455 **physicalDevice** attribute, if the agent implements such a device MIB. 1456 1457 Implementations MAY, though they NEED NOT, include additional values in the job's **imJobStateReasons1** object to indicate the progress of the job, such as adding the 1458 1459 **jobPrinting** value to indicate when the device is actually making marks on a medium. 1460

1 1 - 1	
1461	processingStopped(6),
1462	The job has stopped while processing for any number of reasons and will return to the
1463	processing state as soon as the reasons are no longer present.
1464	
1465	The job's jmJobStateReasons1 object and/or the job's jobStateReasonsN (N=24)
1466	attributes MAY indicate why the job has stopped processing. For example, if the output
1467	device is stopped, the deviceStopped value MAY be included in the job's
1468	jmJobStateReasons1 object.
1469	
1470	NOTE - When an output device is stopped, the device usually indicates its condition in
1471	human readable form at the device. The management application can obtain more
1472	complete device status remotely by querying the appropriate device MIB using the job's
1473	deviceIndex attribute(s), if the agent implements such a device MIB
1474	
1475	canceled(7),
1476	A client has canceled the job and the job is either: (1) in the process of being terminated by
1477	the server or device or (2) has completed terminating. The job's jmJobStateReasons1
1478	object SHOULD contain either the canceledByUser or canceledByOperator value.
1479	sofer she che contain chief the current aby eser of current aby operator value.
1480	aborted(8),
1481	The job has been aborted by the system, usually while the job was in the processing or
1482	processingStopped state.
1483	processingstopped suite.
1485	completed(9)
1485	The job has completed successfully or with warnings or errors after processing and all of
1485	the media have been successfully stacked in the appropriate output bin(s). The job's
1480	jmJobStateReasons1 object SHOULD contain one of: completedSuccessfully,
1488	completedWithWarnings, or completedWithErrors values."
1489	REFERENCE
1490	"This is a type 2 enumeration. See Section 3.6.1.2."
1491	SYNTAX INTEGER {
1492	unknown(2),
1492	pending(3),
1494	pending(3), pendingHeld(4),
1495	processing(5),
1495	processing(5), processingStopped(6),
1490	canceled(7),
1497 1498	aborted(8),
1498	completed(9)
	completed(9)
1500 1501	j
1502	
1503	JmAttributeTypeTC ::= TEXTUAL-CONVENTION
1503	STATUS current
1505	DESCRIPTION
1506	"The type of the attribute which identifies the attribute.
1507	

1508 1509 1510 1511 1512	the attribute SHALL be represented us	ns, each description indicates whether the useful value of sing the jmAttributeValueAsInteger or the y the initial tag: ' INTEGER: ' or ' OCTETS: ',
1512 1513 1514 1515 1516		nenter a choice of useful values of either an integer, an ng on implementation. These attributes are indicated with gs.
1510 1517 1518 1519 1520 1521	(see mediumConsumed(171)). These	ects at the same time to represent a pair of useful values e attributes are indicated with ' INTEGER: ' AND eGroup for the descriptions of these two MANDATORY
1521 1522 1523 1524 1525		buped logically with values assigned in groups of 20, so d in the future and assigned a value that is part of their
1525 1526 1527	NOTE: No attribute name exceeds 31	characters.
1527 1528 1529	The standard attribute types defined at	the time of completion of the specification are:
1529 1530 1531	jmAttributeTypeIndex	Datatype
1532 1533 1534 1535	other(1),	Integer32(-22147483647) AND/OR OCTET STRING(SIZE(063))
1536 1537 1538	INTEGER: and/or OCTETS: approved and registered with IA	An attribute that is not in the list and/or that has not been
1539 1540 1541 1542	++++++++++++++++++++++++++++++++++++++	******
1543 1544 1545	+ The following attributes specify th	ne state of a job. ++++++++++++++++++++++++++++++++++++
1546 1547 1548 1549 1550		JmJobStateReasons2TC tion about the job's current state that augments the cription under the JmJobStateReasons1TC textual-
1550 1551 1552 1553 1554 1555		JmJobStateReasons3TC tion about the job's current state that augments the cription under JmJobStateReasons1TC textual-

1556	jobStateReasons4(5), JmJobStateReasons4TC
1557	INTEGER: Additional information about the job's current state that augments the
1558	jmJobState object. See the description under JmJobStateReasons1TC textual-
1559	convention.
1560	
1561	processingMessage(6), JmUTF8StringTC(SIZE(063))
1562	OCTETS: MULTI-ROW: A coded character set message that is generated by the server
1563	or device during the processing of the job as a simple form of processing log to show
1564	progress and any problems.
1565	progress and any problems.
	There is no restriction for the same massage convering in multiple rows
1566	There is no restriction for the same message occurring in multiple rows.
1567	
1568	jobCodedCharSet(7), CodedCharSet
1569	INTEGER: The MIBenum identifier of the coded character set that the agent is using to
1570	represent coded character set objects and attributes of type 'JmJobStringTC'. These
1571	coded character set objects and attributes are either: (1) supplied by the job submitting
1572	client or (2) defaulted by the server or device when omitted by the job submitting client.
1573	The agent SHALL represent these objects and attributes in the MIB either (1) in the coded
1574	character set as they were submitted or (2) MAY convert the coded character set to
1575	another coded character set or encoding scheme as identified by the jobCodedCharSet
1576	attribute.
1577	
1578	These MIBenum values are assigned by IANA [IANA-charsets] when the coded character
1579	sets are registered. The coded character set SHALL be one of the ones registered with
1580	IANA [IANA] and the enum value uses the CodedCharSet textual-convention from the
1581	Printer MIB. See the JmJobStringTC textual-convention.
1582	
1583	If the agent does not know what coded character set was used by the job submitting client,
1584	the agent SHALL return the 'unknown(2)' value for the jobCodedCharSet attribute for
1585	the job. See Section 3.5.2, entitled "JmJobStringTC' for text generated by the job
1586	submitter'.
1587	
1588	
1589	
1590	***********
1591	+ Job Identification attributes
1592	+
1593	+ The following attributes help an end user, a system
1594	+ operator, or an accounting program identify a job.
1595	
1596	
1597	
1598	
1599	jobAccountName(21), JmJobStringTC(SIZE(063))
1600	OCTETS: Arbitrary binary information which MAY be coded character set data or
1601	encrypted data supplied by the submitting user for use by accounting services to allocate
1602	or categorize charges for services provided, such as a customer account name or number.
1603	or eace of the charges for services provided, such as a customer account name of number.
1604	NOTE: This attribute NEED NOT be printable characters.
1001	TOTE. This during to The Printable characters.

1605	
1606	serverAssignedJobName(22), JmJobStringTC(SIZE(063))
1607	OCTETS: Configuration 3 only: The human readable string name, number, or ID of the
1608	job as assigned by the server that submitted the job to the device that the agent is
1609	providing access to with this MIB.
1610	
1611	NOTE - This attribute is intended for enabling a user to find his/her job that a server
1612	submitted to a device when either the client does not support the jmJobSubmissionID or
1613	the server does not pass the jmJobSubmissionID through to the device.
1614	
1615	jobName(23), JmJobStringTC(SIZE(063))
1616	OCTETS: The human readable string name of the job as assigned by the submitting user
1617	to help the user distinguish between his/her various jobs. This name does not need to be
1618	unique.
1619	1
1620	This attribute is intended for enabling a user or the user's application to convey a job name
1621	that MAY be printed on a start sheet, returned in a query result, or used in notification or
1622	logging messages.
1623	
1624	In order to assist users to find their jobs for job submission protocols that don't supply a
1625	jmJobSubmissionID , the agent SHOULD maintain the jobName attribute for the time
1626	specified by the jmGeneralJobPersistence object, rather than the (shorter)
1627	jmGeneralAttributePersistence object.
1628	0 J
1629	If this attribute is not specified when the job is submitted, no job name is assumed, but
1630	implementation specific defaults are allowed, such as the value of the documentName
1631	attribute of the first document in the job or the fileName attribute of the first document in
1632	the job.
1633	
1634	The jobName attribute is distinguished from the jobComment attribute, in that the
1635	jobName attribute is intended to permit the submitting user to distinguish between
1636	different jobs that he/she has submitted. The jobComment attribute is intended to be free
1637	form additional information that a user might wish to use to communicate with
1638	himself/herself, such as a reminder of what to do with the results or to indicate a different
1639	set of input parameters were tried in several different job submissions.
1640	
1641	jobServiceTypes(24), JmJobServiceTypesTC
1642	INTEGER: Specifies the type(s) of service to which the job has been submitted (print,
1643	fax, scan, etc.). The service type is bit encoded with each job service type so that more
1644	general and arbitrary services can be created, such as services with more than one
1645	destination type, or ones with only a source or only a destination. For example, a job
1646	service might scan, faxOut, and print a single job. In this case, three bits would be set in
1647	the jobServiceTypes attribute, corresponding to the hexadecimal values: $0x8 + 0x20 + 0x^2$
1648	0x4 , respectively, yielding: 0x2C .
1649	
1650	Whether this attribute is set from a job attribute supplied by the job submission client or is
1651	set by the recipient job submission server or device depends on the job submission
1652	protocol. This attribute SHALL be implemented if the server or device has other types in
1653	addition to or instead of printing.

One of the purposes of this attribute is to permit a requester to filter out jobs that are not of interest. For example, a printer operator may only be interested in jobs that include printing.

jobSourceChannelIndex(25),

1654 1655

1656

1657

1658 1659

1660 1661

1662

1663

1664

1665

1666 1667

1668 1669

1670

1671

1672 1673

1674 1675

1676 1677

1678

1679

1680 1681

1682 1683

1684 1685

1686

1687 1688

1689 1690

1691 1692

1693

1694

1695 1696 1697

1698

1699

1700

1701

1702

Integer32(0..2147483647) INTEGER: The index of the row in the associated Printer MIB[print-mib] of the channel which is the source of the print job.

jobSourcePlatformType(26),

INTEGER: The source platform type of the immediate upstream submitter that submitted the job to the server (configuration 2) or device (configuration 1 and 3) to which the agent is providing access. For configuration 1, this is the type of the client that submitted the job to the device; for configuration 2, this is the type of the client that submitted the job to the server; and for configuration 3, this is the type of the server that submitted the job to the device.

submittingServerName(27),

JmJobStringTC(SIZE(0..63)) OCTETS: For configuration 3 only: The administrative name of the server that submitted the job to the device.

submittingApplicationName(28),

OCTETS: The name of the client application (not the server in configuration 3) that submitted the job to the server or device.

jobOriginatingHost(29),

OCTETS: The name of the client host (not the server host name in configuration 3) that submitted the job to the server or device.

deviceNameRequested(30),

OCTETS: The administratively defined coded character set name of the target device requested by the submitting user. For configuration 1, its value corresponds to the Printer MIB[print-mib]: prtGeneralPrinterName object. For configuration 2 and 3, its value is the name of the logical or physical device that the user supplied to indicate to the server on which device(s) they wanted the job to be processed.

queueNameRequested(31),

OCTETS: The administratively defined coded character set name of the target queue requested by the submitting user. For configuration 1, its value corresponds to the queue in the device for which the agent is providing access. For configuration 2 and 3, its value is the name of the queue that the user supplied to indicate to the server on which device(s) they wanted the job to be processed.

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

physicalDevice(32),

hrDeviceIndex AND/OR JmUTF8StringTC(SIZE(0..63))

JmJobStringTC(SIZE(0..63))

JmJobStringTC(SIZE(0..63))

JmJobSourcePlatformTypeTC

Aug 8, 1997

JmJobStringTC(SIZE(0..63))

JmJobStringTC(SIZE(0..63))

1703 1704 1705 1706	INTEGER: MULTI-ROW: The index of the requested/used, such as the Printer MIB[print value. See the Host Resources MIB[hr-mib]	nt-mib]. This value is an hrDeviceIndex
1707 1708	AND/OR	
1709 1710	OCTETS: MULTI-ROW: The name of the	physical device to which the job is assigned.
1711 1712 1713	numberOfDocuments(33), INTEGER: The number of documents in th	Integer32(-22147483647) is job.
1714	fileName(34),	JmJobStringTC(SIZE(063))
1715 1716	OCTETS: MULTI-ROW: The coded chara document.	
1717 1718 1710	There is no restriction on the same file name	e occurring in multiple rows.
1719	1	
1720 1721	documentName(35), OCTETS: MULTI-ROW: The coded chara	JmJobStringTC (SIZE (063)) acter set name of the document.
1722 1723 1724	There is no restriction on the same documen	t name occurring in multiple rows.
1724	ichCommont(26)	Im Joh String TC (SIZE (0. 62))
	jobComment(36),	JmJobStringTC(SIZE(063))
1726	OCTETS: An arbitrary human-readable cod	
1727	submitting user of the job submitting application of the submittin	ation program for any purpose. For example,
1728	a user might indicate what he/she is going to	
1729	submitting application program might indica	ate now the document was produced.
1730		
1731	The jobComment attribute is not intended t	to be a name; see the jobName attribute.
1732		
1733	documentFormatIndex(37),	Integer32(02147483647)
1734	INTEGER: MULTI-ROW: The index in the	ne prtInterpreterTable in the Printer
1735	MIB[print-mib] of the page description lang	uage (PDL) or control language interpreter
1736	that this job requires/uses. A document or a	job MAY use more than one PDL or control
1737	language.	•
1738		
1739	NOTE - As with all intensive attributes when	re multiple rows are allowed, there SHALL be
1740	only one distinct row for each distinct interp	
1741		·····, ····· ···· ···· ···· ····
1742	NOTE - This attribute type is intended to be	used with an agent that implements the
1743	Printer MIB and SHALL not be used if the a	
1744	Such an agent SHALL use the documentFo	
1745	Such an agoin sin ill ase the ascantenti	
1746	documentFormat(38),	PrtInterpreterLangFamilyTC
1747		AND/OR
1748		OCTET STRING(SIZE(063))
1749	INTEGER MULTI-ROW The interpreter	language family corresponding to the Printer
1750	MIB[print-mib] prtInterpreterLangFamily	
1751	document or a job MAY use more than one	
1701	document of a job wirst use more than one	i DD of control lunguage.

Aug 8, 1997

1752 1753 AND/OR 1754 1755 OCTETS: MULTI-ROW: The document format registered as a media type[iana-media-1756 types], i.e., the name of the MIME content-type/subtype. Examples: 1757 'application/postscript', 'application/vnd.hp-PCL', and 'application/pdf' 1758 1759 1760 1761 + Job Parameter attributes 1762 + 1763 + The following attributes represent input parameters 1764 + supplied by the submitting client in the job submission 1765 + protocol. 1766 1767 1768 jobPriority(50), Integer32(1..100) INTEGER: The priority for scheduling the job. It is used by servers and devices that 1769 1770 employ a priority-based scheduling algorithm. 1771 1772 A higher value specifies a higher priority. The value **1** is defined to indicate the lowest 1773 possible priority (a job which a priority-based scheduling algorithm SHALL pass over in 1774 favor of higher priority jobs). The value **100** is defined to indicate the highest possible 1775 priority. Priority is expected to be evenly or 'normally' distributed across this range. The 1776 mapping of vendor-defined priority over this range is implementation-specific. 1777 1778 jobProcessAfterDateAndTime(51), **DateAndTime** (SNMPv2-TC) 1779 OCTETS: The calendar date and time of day after which the job SHALL become a 1780 candidate to be scheduled for processing. If the value of this attribute is in the future, the 1781 server SHALL set the value of the job's **jmJobState** object to **pendingHeld** and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons1 object. When the 1782 specified date and time arrives, the server SHALL remove the jobProcessAfterSpecified 1783 1784 bit value from the job's jmJobStateReasons1 object and, if no other reasons remain, 1785 SHALL change the job's **jmJobState** object to **pending**. 1786 1787 jobHold(52), **JmBooleanTC** INTEGER: If the value is 'true(4)', a client has explicitly specified that the job is to be 1788 held until explicitly released. Until the job is explicitly released by a client, the job SHALL 1789 1790 be in the **pendingHeld** state with the **jobHoldSpecified** value in the 1791 jmJobStateReasons1 attribute. 1792 1793 jobHoldUntil(53), JmJobStringTC(SIZE(0..63)) 1794 OCTETS: The named time period during which the job SHALL become a candidate for processing, such as 'evening', 'night', 'weekend', 'second-shift', 'third-shift', etc., as 1795 defined by the system administrator. See IPP [ipp-model] for the standard keyword values. Until that time period arrives, the job SHALL be in the **pendingHeld** state with 1796 1797 1798 the jobHoldUntilSpecified value in the jmJobStateReasons1 object. The value 'no-1799 hold' SHALL indicate explicitly that no time period has been specified; the absence of this 1800 attribute SHALL indicate implicitly that no time period has been specified.

1801	
1802	outputBin(54), Integer32(02147483647)
1803	AND/OR
1804	JmJobStringTC(SIZE(063))
1805	INTEGER: MULTI-ROW: The output subunit index in the Printer MIB[print-mib]
1805	INTEOLK. MOLTI-KOW. The output subunit index in the Finite Mid[print-into]
1800	AND/OR
	AND/OK
1808	OCTETS, the newspaper provides (as presented as ASCH disite) of the estimation to estimate
1809	OCTETS: the name or number (represented as ASCII digits) of the output bin to which
1810	all or part of the job is placed in.
1811	
1812	sides(55), Integer32(-22)
1813	INTEGER: MULTI-ROW: The number of sides, '1' or '2', that any document in this job
1814	requires/used.
1815	
1816	finishing(56), JmFinishingTC
1817	INTEGER: MULTI-ROW: Type of finishing that any document in this job requires/used.
1818	
1819	
1820	*****************
1821	+ Image Quality attributes (requested and consumed)
1822	+
1823	+ For devices that can vary the image quality.
1824	+++++++++++++++++++++++++++++++++++++++
1825	
1826	printQualityRequested(70), JmPrintQualityTC
1827	INTEGER: MULTI-ROW: The print quality selection requested for a document in the
1828	job for printers that allow quality differentiation.
1829	J F
1830	printQualityUsed(71), JmPrintQualityTC
1831	INTEGER: MULTI-ROW: The print quality selection actually used by a document in the
1832	job for printers that allow quality differentiation.
1833	Job for printers that ano it quality afferentiation.
1834	nrinterResolutionRequested(72)
1834 1835	printerResolutionRequested(72), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for
1835	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for
1835 1836	
1835 1836 1837	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection.
1835 1836 1837 1838	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC
1835 1836 1837 1838 1839	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job
1835 1836 1837 1838 1839 1840	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC
1835 1836 1837 1838 1839 1840 1841	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job for printers that support resolution selection.
1835 1836 1837 1838 1839 1840 1841 1842	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job for printers that support resolution selection. tonerEcomonyRequested(74), JmTonerEconomyTC
1835 1836 1837 1838 1839 1840 1841 1842 1843	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job for printers that support resolution selection. tonerEcomonyRequested(74), JmTonerEconomyTC INTEGER: MULTI-ROW: The toner economy selection requested for documents in the
1835 1836 1837 1838 1839 1840 1841 1842 1843 1844	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job for printers that support resolution selection. tonerEcomonyRequested(74), JmTonerEconomyTC
1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job for printers that support resolution selection. tonerEcomonyRequested(74), JmTonerEconomyTC INTEGER: MULTI-ROW: The toner economy selection requested for documents in the job for printers that allow toner economy differentiation.
1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job for printers that support resolution selection. tonerEcomonyRequested(74), JmTonerEconomyTC INTEGER: MULTI-ROW: The toner economy selection requested for documents in the job for printers that allow toner economy differentiation. tonerEcomonyUsed(75), JmTonerEconomyTC
1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846 1847	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job for printers that support resolution selection. tonerEcomonyRequested(74), JmTonerEconomyTC INTEGER: MULTI-ROW: The toner economy selection requested for documents in the job for printers that allow toner economy differentiation. tonerEcomonyUsed(75), JmTonerEconomyTC INTEGER: MULTI-ROW: The toner economy selection requested for documents in the job for printers that allow toner economy differentiation.
1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846 1847 1848	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job for printers that support resolution selection. tonerEcomonyRequested(74), JmTonerEconomyTC INTEGER: MULTI-ROW: The toner economy selection requested for documents in the job for printers that allow toner economy differentiation. tonerEcomonyUsed(75), JmTonerEconomyTC
1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846 1847	OCTETS: MULTI-ROW: The printer resolution requested for a document in the job for printers that support resolution selection. printerResolutionUsed(73), JmPrinterResolutionTC OCTETS: MULTI-ROW: The printer resolution actually used by a document in the job for printers that support resolution selection. tonerEcomonyRequested(74), JmTonerEconomyTC INTEGER: MULTI-ROW: The toner economy selection requested for documents in the job for printers that allow toner economy differentiation. tonerEcomonyUsed(75), JmTonerEconomyTC INTEGER: MULTI-ROW: The toner economy selection requested for documents in the job for printers that allow toner economy differentiation.

Job Monitoring MIB, V0.85 Aug 8, 1997

1850 1851 1852 1853 1854 1855	tonerDensityRequested(76),Integer32(-2100)INTEGER: MULTI-ROW: The toner density requested for a document in this job for devices that can vary toner density levels. Level 1 is the lowest density and level 100 is the highest density level. Devices with a smaller range, SHALL map the 1-100 range evenly onto the implemented range.
1856 1857 1858 1859 1860 1861 1862	tonerDensityUsed(77),Integer32(-2100)INTEGER: MULTI-ROW: The toner density used by documents in this job for devices that can vary toner density levels. Level 1 is the lowest density and level 100 is the highest density level. Devices with a smaller range, SHALL map the 1-100 range evenly onto the implemented range.
1863 1863 1864 1865 1866	++++++++++++++++++++++++++++++++++++++
1867 1868	 + applications to show an indication of relative progress + to users.
1869 1870 1871	++++++++++++++++++++++++++++++++++++++
1872 1873	INTEGER: The number of copies of the entire job that are to be produced.
1874 1875 1876	jobCopiesCompleted(91), Integer32(-22147483647) INTEGER: The number of copies of the entire job that have been completed so far.
1877 1878 1879 1880 1881	documentCopiesRequested(92), Integer32(-22147483647) INTEGER: The total count of the number of document copies requested for the job as a whole. If there are documents A, B, and C, and document B is specified to produce 4 copies, the number of document copies requested is 6 for the job.
1882 1883 1884	This attribute SHALL be used only when a job has multiple documents. The jobCopiesRequested attribute SHALL be used when the job has only one document.
1885 1886 1887 1888 1889 1890	documentCopiesCompleted(93),Integer32(-22147483647)INTEGER: The total count of the number of document copies completed so far for the job as a whole. If there are documents A, B, and C, and document B is specified to produce 4 copies, the number of document copies starts a 0 and runs up to 6 for the job as the job processes.
1891 1892 1893	This attribute SHALL be used only when a job has multiple documents. The jobCopiesCompleted attribute SHALL be used when the job has only one document.
1894 1895 1896 1897 1898	jobKOctetsTransferred(94), Integer32(-22147483647) INTEGER: The number of K (1024) octets transferred to the server or device to which the agent is providing access. This count is independent of the number of copies of the job or documents that will be produced, but it is only a measure of the number of bytes transferred to the server or device.

1899	
1900	The agent SHALL round the actual number of octets transferred up to the next higher K.
1901	Thus 0 octets SHALL be represented as '0', 1-1024 octets SHALL BE represented as '1',
1902	1025-2048 SHALL be '2', etc. When the job completes, the values of the
1902	jmJobKOctetsRequested object and the jobKOctetsTransferred attribute SHALL be
1904	equal.
1905	
1906	NOTE - The jobKOctetsTransferred can be used with the jmJobKOctetsRequested
1907	object in order to produce a relative indication of the progress of the job for agents that do
1908	not implement the jmJobKOctetsProcessed object.
1909	
1910	
1911	**********
1912	+ Impression attributes
1912	•
	+ . For a mainticht on immersion is the marking of the
1914	+ For a print job, an impression is the marking of the
1915	+ entire side of a sheet. Two-sided processing involves two
1916	+ impressions per sheet. Two-up is the placement of two
1917	+ logical pages on one side of a sheet and so is still a
1918	+ single impression. See also jmJobImpressionsRequested and
1919	+ jmJobImpressionsCompleted objects in the jmJobTable.
1920	+++++++++++++++++++++++++++++++++++++++
1921	
1922	impressionsSpooled(110), Integer32(-22147483647)
1923	INTEGER: The number of impressions spooled to the server or device for the job so far.
1924	
1925	impressionsSentToDevice(111), Integer32(-22147483647)
1926	INTEGER: The number of impressions sent to the device for the job so far.
1927	invitibility. The number of impressions sent to the device for the job so fur.
1928	impressionsInterpreted(112), Integer32(-22147483647)
1928	
	INTEGER: The number of impressions interpreted for the job so far.
1930	
1931	impressionsCompletedCurrentCopy(113), Integer32(-22147483647)
1932	INTEGER: The number of impressions completed by the device for the current copy of
1933	the current document so far. For printing, the impressions completed includes
1934	interpreting, marking, and stacking the output. For other types of job services, the
1935	number of impressions completed includes the number of impressions processed.
1936	
1937	This value SHALL be reset to 0 for each document in the job and for each document
1938	copy.
1939	
1940	fullColorImpressionsCompleted(114), Integer32(-22147483647)
1941	INTEGER: The number of full color impressions completed by the device for this job so
1942	far. For printing, the impressions completed includes interpreting, marking, and stacking
1942	
	the output. For other types of job services, the number of impressions completed includes
1944	the number of impressions processed. Full color impressions are typically defined as those
1945	requiring 3 or more colorants, but this MAY vary by implementation.
1946	

1947	highlightColorImpressionsCompleted(115), Integer32(-2
1948	2147483647)
1949	INTEGER: The number of highlight color impressions completed by the device for this
1950	job so far. For printing, the impressions completed includes interpreting, marking, and
1951	stacking the output. For other types of job services, the number of impressions completed
1952	includes the number of impressions processed. Highlight color impressions are typically
1953	defined as those requiring black plus one other colorant, but this MAY vary by
1955	implementation.
1955	Implementation.
1955	
1950	
1958	+ Page attributes
1959	
1960	+ A page is a logical page. Number up can impose more than
1961	+ one page on a single side of a sheet. Two-up is the
1962	+ placement of two logical pages on one side of a sheet so
1963	+ that each side counts as two pages.
1964	****************
1965	
1966	pagesRequested(130), Integer32(-22147483647)
1967	INTEGER: The number of logical pages requested by the job to be processed.
1968	
1969	pagesCompleted(131), Integer32(-22147483647)
1970	INTEGER: The number of logical pages completed for this job so far.
1971	
1972	For implementations where multiple copies are produced by the interpreter with only a
1972	single pass over the data, the final value SHALL be equal to the value of the
1973	pagesRequested object. For implementations where multiple copies are produced by the
1975	interpreter by processing the data for each copy, the final value SHALL be a multiple of
1976	the value of the pagesRequested object.
1970	the value of the pageskequested object.
1978	NOTE - See the impressionsCompletedCurrentCopy and
1978	pagesCompletedCurrentCopy attributes for attributes that are reset on each document
1979	
	copy.
1981	NOTE. The manual data data is the second south the manual D emocrated at the test
1982	NOTE - The pagesCompleted object can be used with the p agesRequested object to
1983	provide an indication of the relative progress of the job, provided that the multiplicative factor is taken into account for some implementations of multiple copies.
1984	factor is taken into account for some implementations of multiple copies.
1985	
1986	pagesCompletedCurrentCopy(132), Integer32(-22147483647)
1987	INTEGER: The number of logical pages completed for the current copy of the document
1988	so far. This value SHALL be reset to 0 for each document in the job and for each
1989	document copy.
1990	
1991	
1992	***********
1993	+ Sheet attributes
1994	+
1995	+ The sheet is a single piece of a medium, whether printing

1996	+ on one or both sides.	
1997	+++++++++++++++++++++++++++++++++++++++	+++++++++++++++++++++++++++++++++++++++
1998		
1999	sheetsRequested(150),	Integer32(-22147483647)
2000	INTEGER: The number of medium sheets	
2000	INTEGER. The number of medium sheets	requested to be processed for this job.
2002	sheetsCompleted(151),	Integer32(-22147483647)
2002		that have completed marking and stacking for
2003	the entire job so far whether those sheets ha	
2004	the entire job so far whether those sheets ha	we been processed on one side of on bour.
	-h	T
2006	sheetsCompletedCurrentCopy(152),	Integer32(-22147483647)
2007	INTEGER: The number of medium sheets	that have completed marking and stacking for
2008	the current copy of a document in the job so	o far whether those sheets have been processed
2009	on one side or on both.	
2010		
2011		to 0 as each document in the job starts being
2012	processed and for each document copy as it	starts being processed.
2013		
2014		
2015	*++++++++++++++++++++++++++++++++++++++	+++++++++++++++++++++++++++++++++++++++
2016	+ Resources attributes (requested and consum	ed)
2017	+	
2018	+ Pairs of these attributes can be used by monit	itoring
2019	+ applications to show an indication of relative	e usage to
2020	+ users.	
2021	***********	*****
2022		
2023	mediumRequested(170),	JmMediumTypeTC
2024		AND/OR
2025		JmJobStringTC(SIZE(063))
2026	INTEGER: MULTI-ROW: The type	······································
2027	AND/OR	
2028	OCTETS: the name of the medium that is r	required by the job
2029		equiled by the job.
2030	mediumConsumed(171),	Integer32(-22147483647)
2030	meurum consumeu (171),	AND
2032		JmJobStringTC(SIZE(063))
2032	INTEGER: The number of sheets	3113005tring1C(512E(003))
2033	AND	
2035	OCTETS: MULTI-ROW: the name of the	madium that has been consumed so far
2035		
2030	whether those sheets have been processed of	on one side of on bour.
	This attailante CIIAII, house hoth Intercor??	and OCTET STDING (nonresponded on
2038	This attribute SHALL have both Integer32	and OCTET STRING (represented as
2039	JmJobStringTC) values.	
2040		
2041	colorantRequested(172),	Integer32(-22147483647)
2042		AND/OR
2043		JmJobStringTC(SIZE(063))
2044	INTEGER: MULTI-ROW: The index (pr	tiviarkerColorantIndex) in the Printer

2045 2046	MIB[print-mib] AND/OR	
2040	OCTETS: the name of the colorant req	uested.
2048	1	
2049	colorantConsumed(173),	Integer32(-22147483647)
2050		AND/OR
2051		JmJobStringTC(SIZE(063))
2052	INTEGER: MULTI-ROW: The index	(prtMarkerColorantIndex) in the Printer
2053	MIB[print-mib]	(F
2054	AND/OR	
2055	OCTETS: the name of the colorant cons	sumed
2055	OCTETS: the number of the colorant com	Junica.
2050		
2057		
2058	++++++++++++++++++++++++++++++++++++++	
2059	+ Time attributes (set by server or device)	
	+	a got by the
2061	+ This section of attributes are ones that an	
2062	+ server or device that accepts jobs. Two f	
2063	+ provided. Each form is represented in a	
2064	+ See section 3.1.2 and section 3.1.3 for the	
2065	+ conformance requirements for time attri	
2066	+ monitoring applications, respectively. The	ne two forms are:
2067	+	
2068	+ 'DateAndTime' is an 8 or 11 octet binary	
2069	+ month, day, hour, minute, second, deci-s	
2070	+ optional offset from UTC. See SNMPv2-	TC [SMIv2-TC].
2071	+	
2072	+ NOTE: 'DateAndTime' is not printable c	haracters; it is
2073	+ binary.	
2074	+	
2075	+ 'JmTimeStampTC' is the time of day me	asured in the number of
2076	+ seconds since the system was booted.	
2077	+++++++++++++++++++++++++++++++++++++++	+++++++++++++++++++++++++++++++++++++++
2078		
2079	jobSubmissionToServerTime(190),	JmTimeStampTC
2080		AND/OR
2081		DateAndTime
2082	INTEGER: Configuration 3 only: The	time
2083	AND/OR	
2084	OCTETS: the date and time that the jol	o was submitted to the server (as distinguished
2085	from the device which uses jobSubmissi	
2086		
2087	jobSubmissionTime(191),	JmTimeStampTC
2088	•	AND/OR
2089		DateAndTime
2090	INTEGER: Configurations 1, 2, and 3:	
2091	AND/OR	
2092		b was submitted to the server or device to which
2093	the agent is providing access.	
	-	

Aug 8, 1997

2095		
2096		
2097	jobStartedBeingHeldTime(192),	JmTimeStampTC
2098		AND/OR
2099		DateAndTime
2100	INTEGER: The time	
2101	AND/OR	
2102		e job last entered the pendingHeld state. If the job
2102		state, then the value SHALL be '0' or the attribute
2103	SHALL not be present in the table.	state, then the value STITLE be 0 of the attribute
2104	STRALL not be present in the table.	
2105 2106	jobStartedProcessingTime(193),	JmTimeStampTC
	jobstarteur rocessing rime(195),	AND/OR
2107		
2108		DateAndTime
2109	INTEGER: The time	
2110	AND/OR	
2111	OCTETS: the date and time that the	e job started processing.
2112		, _, _,
2113	jobCompletedTime(194),	JmTimeStampTC
2114		AND/OR
2115		DateAndTime
2116	INTEGER: The time	
2117	AND/OR	
2118	OCTETS: the date and time that the	e job entered the completed , canceled , or aborted
2119	state.	J L / /
2120		
2121	jobProcessingCPUTime(195)	Integer32(-22147483647)
2122	UNITS 'seconds'	g•=•=(i i i i i i i i i i i i i i i i
2123		he in seconds that the job has been in the processing
2124		gStopped state, that elapsed time SHALL not be
2125		ocessingCPUTime value SHOULD be relatively
2125	repeatable when the same job is proc	
2120	repeatable when the same job is proc	cessed again on the same device.
2128	REFERENCE	
2120		echanism' for a description of this textual-convention
2129	and its use in the jmAttributeTable .	echanishi for a description of this textual-convention
2130	and its use in the jinAtti ibute i able .	
	This is a true 2 commention - Sac Section	2 6 1 2 "
2132	This is a type 2 enumeration. See Section	3.0.1.2.
2133	SYNTAX INTEGER {	
2134	other(1),	
2135	unknown(2),	
2136	jobStateReasons2(3),	
2137	jobStateReasons3(4),	
2138	jobStateReasons4(5),	
2139	processingMessage(6),	
2140	jobCodedCharSet(7),	
2141	•	
2142	jobAccountName(21),	

2094

2143	serverAssignedJobName(22),
2144	jobName(23),
	5
2145	jobServiceTypes(24),
2146	jobSourceChannelIndex(25),
2147	jobSourcePlatformType(26),
2148	submittingServerName(27),
2149	0
	submittingApplicationName(28),
2150	jobOriginatingHost(29),
2151	deviceNameRequested(30),
2152	queueNameRequested(31),
2153	physicalDevice(32),
2154	numberOfDocuments(33),
2155	
	fileName(34),
2156	documentName(35),
2157	jobComment(36),
2158	documentFormatIndex(37),
2159	documentFormat(38),
2160	
2160	jobPriority(50),
2162	jobProcessAfterDateAndTime(51),
2163	jobHold(52),
2164	jobHoldUntil(53),
2165	outputBin(54),
2166	sides(55),
2167	finishing(56),
	minsining(50),
2168	
2169	printQualityRequested(70),
2170	printQualityUsed(71),
2171	printerResolutionRequested(72),
2172	printerResolutionUsed(73),
2173	tonerEcomonyRequested(74),
2174	tonerEcomonyUsed(75),
2175	tonerDensityRequested(76),
2176	tonerDensityUsed(77),
2177	
2178	jobCopiesRequested(90),
2179	jobCopiesCompleted(91),
2180	documentCopiesRequested(92),
2181	documentCopiesCompleted(93),
2182	jobKOctetsTransferred(94),
2183	
2184	impressionsSpooled(110),
2185	impressionsSentToDevice(111),
2185	
	impressionsInterpreted(112),
2187	impressionsCompletedCurrentCopy(113),
2188	fullColorImpressionsCompleted(114),
2189	highlightColorImpressionsCompleted(115),
2190	
2191	pagesRequested(130),
-1/1	Publicitudica (190),

2192	pagesCompleted(131),
2193	pagesCompletedCurrentCopy(132),
2194	
2195	sheetsRequested(150),
2196	sheetsCompleted(151),
2197	sheetsCompletedCurrentCopy(152),
2198	
2199	mediumRequested(170),
2200	mediumConsumed(171),
2200	colorantRequested(172),
2201	colorantConsumed(173),
2202	colorance on sum cu (175),
2203	ichSuhmissionToSenverTime(100)
	jobSubmissionToServerTime(190),
2205	jobSubmissionTime(191), i=h Stanta dD aina UL1dTime (102)
2206	jobStartedBeingHeldTime(192),
2207	jobStartedProcessingTime(193),
2208	jobCompletedTime(194),
2209	jobProcessingCPUTime(195)
2210	}
2211	
2212	
2213	
2214	
2215	JmJobServiceTypesTC ::= TEXTUAL-CONVENTION
2216	NTATUS current
2216	STATUS current DESCRIPTION
2217	DESCRIPTION
2217 2218	DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The
2217 2218 2219	DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that
2217 2218 2219 2220	DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one
2217 2218 2219 2220 2221	DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service
2217 2218 2219 2220 2221 2222	DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan , faxOut , and print a single job. In this case, three bits would be set in the
2217 2218 2219 2220 2221 2222 2222 2223	DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4,
2217 2218 2219 2220 2221 2222 2223 2223 2224	DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan , faxOut , and print a single job. In this case, three bits would be set in the
2217 2218 2219 2220 2221 2222 2223 2224 2225	DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: $0x8 + 0x20 + 0x4$, respectively, yielding: $0x2C$.
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226	DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job.
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job. One of the purposes of this attribute is to permit a requester to filter out jobs that are not of
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job. One of the purposes of this attribute is to permit a requester to filter out jobs that are not of interest. For example, a printer operator MAY only be interested in jobs that include printing.
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job. One of the purposes of this attribute is to permit a requester to filter out jobs that are not of
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job. One of the purposes of this attribute is to permit a requester to filter out jobs that are not of interest. For example, a printer operator MAY only be interested in jobs that include printing. That is why the attribute is in the job identification category.
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234 2235	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job. One of the purposes of this attribute is to permit a requester to filter out jobs that are not of interest. For example, a printer operator MAY only be interested in jobs that include printing. That is why the attribute is in the job identification category. The following service component types are defined (in hexadecimal) and are assigned a separate
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234 2235 2236	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job. One of the purposes of this attribute is to permit a requester to filter out jobs that are not of interest. For example, a printer operator MAY only be interested in jobs that include printing. That is why the attribute is in the job identification category.
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234 2235 2236 2237	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job. One of the purposes of this attribute is to permit a requester to filter out jobs that are not of interest. For example, a printer operator MAY only be interested in jobs that include printing. That is why the attribute is in the job identification category. The following service component types are defined (in hexadecimal) and are assigned a separate bit value for use with the jobServiceTypes attribute:
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234 2235 2236 2237 2238	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job. One of the purposes of this attribute is to permit a requester to filter out jobs that are not of interest. For example, a printer operator MAY only be interested in jobs that include printing. That is why the attribute is in the job identification category. The following service component types are defined (in hexadecimal) and are assigned a separate bit value for use with the jobServiceTypes attribute: other 0x1
2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234 2235 2236 2237	 DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is represented as an enum that is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C. Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. With either implementation, the agent SHALL return a non-zero value for this attribute indicating the type of the job. One of the purposes of this attribute is to permit a requester to filter out jobs that are not of interest. For example, a printer operator MAY only be interested in jobs that include printing. That is why the attribute is in the job identification category. The following service component types are defined (in hexadecimal) and are assigned a separate bit value for use with the jobServiceTypes attribute:

2240	
2241	unknown 0x2
2242	The job contains some instructions whose type is unknown to the agent.
2243	
2244	print 0x4
2245	The job contains some instructions that specify printing
2246	
2247	scan 0x8
2248	The job contains some instructions that specify scanning
2249	
	for T_ 0-10
2250	faxIn 0x10
2251	The job contains some instructions that specify receive fax
2252	
	for
2253	faxOut 0x20
2254	The job contains some instructions that specify sending fax
2255	
2256	getFile 0x40
2257	The job contains some instructions that specify accessing files or documents
2258	
2259	putFile 0x80
	1
2260	The job contains some instructions that specify storing files or documents
2261	
2262	mailList 0x100
2263	The job contains some instructions that specify distribution of documents using an
2264	electronic mail system."
2265	REFERENCE
2266	"These bit definitions are the equivalent of a type 2 enum except that combinations of them
2267	MAY be used together. See section 3.6.1.2."
2268	SYNTAX INTEGER(02147483647) 31 bits, all but sign bit
2200	STITIA TITICIA(0.2147405047) 51 bits, an out sign bit
2269	
2270	
2270	
2271	
2271	
2272	
2272	Im Joh Stata Dagsons 1 TC TEVTUAL CONVENTION
2273	JmJobStateReasons1TC ::= TEXTUAL-CONVENTION
2274	STATUS current
2275	DESCRIPTION
2276	"The JmJobStateReasonsNTC (<i>N</i> =14) textual-conventions are used with the
2277	jmJobStateReasons1 object and jobStateReasonsN (N=24), respectively, to provide
2278	additional information regarding the current jmJobState object value. These values MAY be
2279	used with any job state or states for which the reason makes sense.
	used with any job state of states for which the reason makes sense.
2280	
2281	NOTE - While values cannot be added to the jmJobState object without impacting deployed
2282	clients that take actions upon receiving jmJobState values, it is the intent that additional
2283	JmJobStateReasonsNTC enums can be defined and registered without impacting such
2284	deployed clients. In other words, the jmJobStateReasons1 object and jobStateReasons N
2285	attributes are intended to be extensible.

Aug 8, 1997

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

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

other

2286 2287

2288

2289

2290

2291

2292 2293

2294

2295

2296

2297

2298 2299

2300

2301 2302

2303

2304 2305

2306

2307

2308

2309 2310

2311

2312 2313

2314 2315

2316

2317 2318

2319

2320

2321

2322 2323

2324

2325

2326

2327 2328

2329 2330

2331

2332

2333

0x1

0x2

0x4

The job state reason is not one of the standardized or registered reasons.

unknown

The job state reason is not known to the agent or is indeterminent.

jobIncoming

The job has been accepted by the server or device, but the server or device is expecting (1) additional operations from the client to finish creating the job and/or (2) is accessing/accepting document data.

iobOutgoing

Configuration 2 only: The server is transmitting the job to the device.

jobHoldSpecified

The value of the job's **jobHold**(52) attribute is TRUE. The job SHALL NOT be a candidate for processing until this reason is removed and there are no other reasons to hold the job.

jobHoldUntilSpecified

The value of the job's **jobHoldUntil**(53) attribute specifies a time period that is still in the future. The job SHALL NOT be a candidate for processing until this reason is removed and there are no other reasons to hold the job.

jobProcessAfterSpecified

The value of the job's jobProcessAfterDateAndTime(51) attribute specifies a time that is still in the future. The job SHALL NOT be a candidate for processing until this reason is removed and there are no other reasons to hold the job.

resourcesAreNotReady

At least one of the resources needed by the job, such as media, fonts, resource objects, etc., is not ready on any of the physical devices for which the job is a candidate. This condition MAY be detected when the job is accepted, or subsequently while the job is pending or processing, depending on implementation.

0x10

0x8

0x20

0x40

0x80

2382

2334	deviceStoppedPartly	0x100
2335		
		which the job is assigned are stopped. If all of
2336	the devices are stopped (or the only device	is stopped), the deviceStopped reason
2337	SHALL be used.	
	STILLE De usea.	
2338		
2339	deviceStopped	0x200
2340	The device(s) to which the job is assigned	is (are all) stopped
	The device(b) to which the job is ussigned	is (are an) stopped.
2341		
2342	jobPrinting	0x400
2343	The output device is marking media. This a	attribute is useful for servers and output devices
2344		g when no marking is happening and then want
2345	to show that marking is now happening or	when the job is in the canceled or aborted
2346	state, but the marking has not yet stopped s	so that impression or sheet counts are still
2347		so that impression of sheet counts are suit
	increasing for the job.	
2348		
2349	jobCanceledByUser	0x800
2350		n unknown user or by a user whose name is the
2351	same as the value of the job's jmJobOwne	r object.
2352		•
2353	jobCanceledByOperator	0x1000
2354	The job was canceled by the operator, i.e.,	by a user whose name is different than the
2355	value of the job's jmJobOwner object.	
2356		
		0.000
2357	abortedBySystem	0x2000
2358	The job was aborted by the system.	
2359	5	
2360	NOTE When the system puts a job into the	he 'shorted' job state, this reason is not needed
		he 'aborted' job state, this reason is not needed.
2361	This reason is needed only when the system	n aborts a job, but, instead of placing the job in
2362	the aborted job state, places the job in the	pendingHeld state, so that a user or operator
2363	can manually try the job again.	
	can manually ify the job again.	
2364		
2365	processingToStopPoint	0x4000
2366		ancel or interrupt the job or the server/device
2367		still performing some actions on the job until a
2368	specified stop point occurs or job terminati	on/cleanup is completed.
2369	1 11 5	1 1
	This was an is was a way of the last of in	
2370		conjunction with the canceled or aborted job
2371	state to indicate that the server/device is sti	ll performing some actions on the job after the
2372		ne of the jobs resources consumed counters
2373		
	may still be incrementing while the job is in	The canceled of aborted job states.
2374		
2375	jobCompletedSuccessfully	0x8000
2376	The job completed successfully.	
	The job completed successfully.	
2377		
2378	jobCompletedWithWarnings	0x10000
2379	The job completed with warnings.	
	The job completed with warnings.	
2380		
2381	jobCompletedWithErrors	0x20000
1201	The ich completed with among (and paggih)	

The job completed with errors (and possibly warnings too).

Aug 8, 1997

[Page 60]

2383 2384 2385 The following additional job state reasons have been added to represent job states that are in 2386 ISO DPA[iso-dpa] and other job submission protocols: 2387 2388 jobPaused **0x40000** The job has been indefinitely suspended by a client issuing an operation to suspend the job 2389 so that other jobs may proceed using the same devices. The client MAY issue an 2390 operation to resume the paused job at any time, in which case the agent SHALL remove 2391 2392 the jobPaused values from the job's jmJobStateReasons1 object and the job is eventually 2393 resumed at or near the point where the job was paused. 2394 2395 jobInterrupted **0x80000** 2396 The job has been interrupted while processing by a client issuing an operation that 2397 specifies another job to be run instead of the current job. The server or device will 2398 automatically resume the interrupted job when the interrupting job completes. 2399 2400 jobRetained **0x100000** 2401 The job is being retained by the server or device with all of the job's document data (and 2402 submitted resources, such as fonts, logos, and forms, if any). Thus a client could issue an 2403 operation to the server or device to either (1) re-do the job (or a copy of the job) on the 2404 same server or device or (2) resubmit the job to another server or device. When a client could no longer re-do/resubmit the job, such as after the document data has been 2405 2406 discarded, the agent SHALL remove the **jobRetained** value from the 2407 jmJobStateReasons1 object." 2408 REFERENCE "These bit definitions are the equivalent of a type 2 enum except that combinations of bits may 2409 2410 be used together. See section 3.6.1.2. The remaining bits are reserved for future 2411 standardization and/or registration." 2412 2413 SYNTAX **INTEGER(0..2147483647)** -- 31 bits, all but sign bit 2414 2415 2416 2417 2418 2419 JmJobStateReasons2TC ::= TEXTUAL-CONVENTION 2420 STATUS current 2421 DESCRIPTION 2422 "This textual-convention is used with the **jobStateReasons2** attribute to provides additional 2423 information regarding the **jmJobState** object. See the description under 2424 JmJobStateReasons1TC for additional information that applies to all reasons. 2425 2426 The following standard values are defined (in hexadecimal) as powers of two, since multiple 2427 values may be used at the same time: 2428 2429 cascaded 0x1 2430 An outbound gateway has transmitted all of the job's job and document attributes and data 2431 to another spooling system.

0.420	
2432	
2433	deletedByAdministrator 0x2
2434	The administrator has deleted the job.
2435	
2436	discardTimeArrived 0x4
2437	The job has been deleted due to the fact that the time specified by the job's job-discard-
2438	time attribute has arrived.
2439	
2440	postProcessingFailed 0x8
2441	The post-processing agent failed while trying to log accounting attributes for the job;
2442	therefore the job has been placed into the completed state with the jobRetained
2443	jmJobStateReasons1 object value for a system-defined period of time, so the
2444	administrator can examine it, resubmit it, etc.
2445	administrator can examine it, resubilit it, etc.
2446	submissionInterrupted 0x10
	1
2447	Indicates that the job was not completely submitted for some unforeseen reason, such as:
2448	(1) the server has crashed before the job was closed by the client, (2) the server or the
2449	document transfer method has crashed in some non-recoverable way before the document
2450	data was entirely transferred to the server, (3) the client crashed or failed to close the job
2451	before the time-out period.
2452	
2453	maxJobFaultCountExceeded 0x20
2454	The job has faulted several times and has exceeded the administratively defined fault count
2455	limit.
2456	
2457	devicesNeedAttentionTimeOut 0x40
2458	One or more document transforms that the job is using needs human intervention in order
2459	for the job to make progress, but the human intervention did not occur within the site-
2460	settable time-out value.
2461	settable time out value.
2462	needsKeyOperatorTimeOut 0x80
2462	
	One or more devices or document transforms that the job is using need a specially trained
2464	operator (who may need a key to unlock the device and gain access) in order for the job to
2465	make progress, but the key operator intervention did not occur within the site-settable
2466	time-out value.
2467	
2468	jobStartWaitTimeOut 0x100
2469	The server/device has stopped the job at the beginning of processing to await human
2470	action, such as installing a special cartridge or special non-standard media, but the job was
2471	not resumed within the site-settable time-out value and the server/device has transitioned
2472	the job to the pendingHeld state.
2473	
2474	jobEndWaitTimeOut 0x200
2475	The server/device has stopped the job at the end of processing to await human action,
2476	such as removing a special cartridge or restoring standard media, but the job was not
2477	resumed within the site-settable time-out value and the server/device has transitioned the
2478	job to the completed state.
2479	Joo to the completed blute.
Δ T I J	

2527

2528

Job Monitoring MIB, V0.85

0x400 2480 jobPasswordWaitTimeOut The server/device has stopped the job at the beginning of processing to await input of the 2481 2482 job's password, but the password was not received within the site-settable time-out value. 2483 2484 deviceTimedOut 0x800 2485 A device that the job was using has not responded in a period specified by the device's 2486 site-settable attribute. 2487 2488 connectingToDeviceTimeOut 0x1000 2489 The server is attempting to connect to one or more devices which may be dial-up, polled, 2490 or queued, and so may be busy with traffic from other systems, but server was unable to 2491 connect to the device within the site-settable time-out value. 2492 2493 transferring 0x2000 2494 The job is being transferred to a down stream server or device. 2495 2496 queuedInDevice 0x4000 2497 The job has been queued in a down stream server or device. 2498 2499 iobCleanup 0x8000 The server/device is performing cleanup activity as part of ending normal processing. 2500 2501 2502 jobPasswordWait 0x20000 2503 The server/device has selected the job to be next to process, but instead of assigning 2504 resources and starting the job processing, the server/device has transitioned the job to the 2505 **pendingHeld** state to await entry of a password (and dispatched another job, if there is 2506 one). 2507 2508 validating **0x40000** 2509 The server/device is validating the job *after* accepting the job. 2510 2511 aueueHeld **0x80000** 2512 The operator has held the entire job set or queue. 2513 2514 **iobProofWait** 0x100000 The job has produced a single proof copy and is in the **pendingHeld** state waiting for the 2515 requester to issue an operation to release the job to print normally, obeying any job and 2516 2517 document copy attributes that were originally submitted. 2518 2519 **heldForDiagnostics** 0x200000 The system is running intrusive diagnostics, so that all jobs are being held. 2520 2521 2522 0x400000 serviceOffLine 2523 The service/document transform is off-line and accepting no jobs. All pending jobs are put 2524 into the pendingHeld state. This could be true if its input is impaired or broken. 2525 2526 0x800000

noSpaceOnServer

There is no room on the server to store all of the job.

2529	pinRequired 0x1000000
2530	The System Administrator settable device policy is (1) to require PINs, and (2) to hold
2531	jobs that do not have a pin supplied as an input parameter when the job was created.
2532	
2533	exceededAccountLimit 0x2000000
2534	The account for which this job is drawn has exceeded its limit. This condition SHOULD
2535	be detected before the job is scheduled so that the user does not wait until his/her job is
2536	scheduled only to find that the account is overdrawn. This condition MAY also occur
2530 2537	while the job is processing either as processing begins or part way through processing.
2538	while the job is processing either as processing begins of part way through processing.
2539	heldForRetry 0x4000000
2540	The job encountered some errors that the server/device could not recover from with its
2541	normal retry procedures, but the error might not be encountered if the job is processed
2542	again in the future. Example cases are phone number busy or remote file system in-
2543	accessible. For such a situation, the server/device SHALL transition the job from the
2544	processing to the pendingHeld , rather than to the aborted state.
2545	
2546	The following values are from the X/Open PSIS draft standard:
2547	
2548	canceledByShutdown 0x8000000
2549	The job was canceled because the server or device was shutdown before completing the
2550	job.
2550	J00.
2552	deviceUnavailable 0x1000000
2552	
	This job was aborted by the system because the device is currently unable to accept jobs.
2554	
2555	wrongDevice 0x2000000
2556	This job was aborted by the system because the device is unable to handle this particular
2557	job; the spooler SHOULD try another device or the user should submit the job to another
2558	device.
2559	
2560	badJob 0x4000000
2561	This job was aborted by the system because this job has a major problem, such as an ill-
2562	formed PDL; the spooler SHOULD not even try another device."
2563	REFERENCE
2564	"These bit definitions are the equivalent of a type 2 enum except that combinations of them may
2565	be used together. See section 3.6.1.2. See the description under JmJobStateReasons1TC and
2566	the jobStateReasons2 attribute."
2567	
2568	SYNTAX INTEGER(02147483647) 31 bits, all but sign bit
2569	S = 1 + 1 + 1 + 1 + 1 + 0 + 1 + 1 + 0 + 0 +
2570	
2571	
2572	
2573	
2574	
2575	JmJobStateReasons3TC ::= TEXTUAL-CONVENTION
2576	STATUS current
2577	DESCRIPTION

2578 2579	"This textual-convention is used with the jobStateReasons3 attribute to provides additional information regarding the jmJobState object. See the description under
2580	JmJobStateReasons1TC for additional information that applies to all reasons.
2581	
2582	The following standard values are defined (in hexadecimal) as powers of two, since multiple
2583	values may be used at the same time:
2584	
2585	jobInterruptedByDeviceFailure 0x1
2586	A device or the print system software that the job was using has failed while the job was
2587	processing. The server or device is keeping the job in the pendingHeld state until an
2588	operator can determine what to do with the job."
2589	REFERENCÉ
2590	"These bit definitions are the equivalent of a type 2 enum except that combinations of them may
2591	be used together. See section 3.6.1.2. The remaining bits are reserved for future
2592	standardization and/or registration. See the description under JmJobStateReasons1TC and the
2593	jobStateReasons3 attribute."
2594	SYNTAX INTEGER(02147483647) 31 bits, all but sign bit
2595	
2596	
2597	
2598	
2599	
2600	JmJobStateReasons4TC ::= TEXTUAL-CONVENTION
2601	STATUS current
2602	DESCRIPTION
2603	"This textual-convention is used in the jobStateReasons4 attribute to provides additional
2604	information regarding the jmJobState object. See the description under
2605	JmJobStateReasons1TC for additional information that applies to all reasons.
2606	
2607	The following standard values are defined (in hexadecimal) as powers of two, since multiple
2608	values may be used at the same time:
2609	·
2610	none yet defined. These bits are reserved for future standardization and/or registration."
2611	REFERENCE
2612	"These bit definitions are the equivalent of a type 2 enum except that combinations of them may
2613	be used together. See section 3.6.1.2. See the description under JmJobStateReasons1TC and
2614	the jobStateReasons4 attribute."
2615	
2616	SYNTAX INTEGER(02147483647) 31 bits, all but sign bit

 2619 The General Group (MANDATORY) 2621 The jmGeneralGroup consists entirely of the jmGeneralTable. 2623 2624 jmGeneral OBJECT IDENTIFIER ::= { jobmonMIBObjects 1 } 2625 2626 jmGeneralTable OBJECT-TYPE 2627 SYNTAX SEQUENCE OF JmGeneralEntry 2628 MAX-ACCESS not-accessible 2629 STATUS current 2630 DESCRIPTION 2631 "The jmGeneralTable consists of information of a general nature that are per-job-set, but are not per-job. See Section 2 entitled "Terminology and Job Model' for the definition of a job set." 2633 REFERENCE 2634 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 	2617			
 The General Group (MANDATORY) The jmGeneralGroup consists entirely of the jmGeneralTable. The jmGeneralTable OBJECT-TYPE SYNTAX SEQUENCE OF ImGeneralEntry MAX-ACCESS not-accessible STATUS current DESCRIPTION The jmGeneralTable consists of information of a general nature that are per-job-set, but are nor per-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." REFERENCE "The MANDATORY-GROUP macro specifies that this group is MANDATORY." ::= { jmGeneralTable consists of information of a general nature that are per-job-set, but are nor per-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." REFERENCE "The MANDATORY-GROUP macro specifies that this group is MANDATORY." ::= { jmGeneralEntry OBJECT-TYPE MGAX-ACCESS not-accessible SYNTAX JmGeneralEntry MAX-ACCESS not-accessible CHATUS current DESCRIPTION ImGeneralIabbetIndex Integer32(132767), jmGeneralJobSetIndex Integer32(132767), jmGeneralJobPersistence Integer32(12147483647), jmGeneralJobSetIndex OBJECT-TYPE SYNTAX Integer32(132767), jmGeneralJobSetIndex OBJECT-TYPE SYNTAX Integer32(132767) jmGeneralJobSetIndex OBJECT-TYPE SYNTAX Integer32(132767) MAX-ACCESS not-accessible DESCRIPTION 	2618	jobmonMIBObjects OBJECT IDENTIFIER ::= { jobmonMIBOBJECT IDENTIFIER ::	MIB 1 }	
 - The jmGeneralGroup consists entirely of the jmGeneralTable. - The jmGeneralGroup consists entirely of the jmGeneralTable. jmGeneralTable OBJECT IDENTIFIER ::= { jobmonMIBObjects 1 } imGeneralTable OBJECT-TYPE SYNTAX SEQUENCE OF JmGeneralEntry MAX-ACCESS not-accessible OESCRIPTION The jmGeneralTable consists of information of a general nature that are per-job-set, but are not per-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." REFERENCE "The MANDATORY-GROUP macro specifies that this group is MANDATORY." ::= { jmGeneralEntry MAX-ACCESS not-accessible SYNTAX JmGeneralEntry mGeneralEntry OBJECT-TYPE SinGeneralEntry OBJECT-TYPE mGeneralIobSetIndex in this table for each job set." Integer32(132767), jmGeneralIobSetIndex Integer32(02147483647), jmGeneralIobSetIndex Integer32(02147483647), jmGeneralIobSetIndex Integer32(02147483647), jmGeneralIobSetIndex Integer32(02147483647), jmGeneralIobSetIndex Integer32(132767), jmGeneralIobSetIndex OBJECT-TYPE SYNTAX Integer32(132767), jmGeneralIobSetIndex OBJECT-TYPE SYNTAX Integer32(132767) jmGeneralIobSetIndex OBJECT-TYPE SYNTAX Integer32(132767) jmGeneralIobSetIndex OBJECT-TYPE JmUTF8StringTC(SIZE(063)) 	2619			
 The jmGeneralGroup consists entirely of the jmGeneralTable. imGeneral OBJECT IDENTIFIER ::= { jobmonMIBObjects 1 } imGeneralTable OBJECT-TYPE SYNTAX SEQUENCE OF JmGeneralEntry MAX-ACCESS not-accessible STATUS current DESCRIPTION "The jmGeneralTable consists of information of a general nature that are per-job-set, but are not per-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." REFERENCE "The jmGeneralEntry OBJECT-TYPE (imGeneralI1) "The jmGeneralEntry (imGeneralI1) "The jmGeneralEntry (imGeneralI1) "The jmGeneralEntry (imGeneralI1) "The jmGeneralEntry (imGeneralI1) "The jmGeneralI1] "The jmGeneralI0058EIIndex in this table for each job set." INDEX { jmGeneralI058EIIndex integer32(02147483647), jmGeneralI058EIIndex integer32(02147483647), jmGeneralI058EIIndex integer32(02147483647), jmGeneralI058EIIndex integer32(02147483647), jmGeneralI058EIIndex integer32(02147483647), jmGeneralI058EIName jmGeneralI058EIName jmGeneralI058EIName integer32(02147483647), jmGeneralI058EIName jmGen	2620	The General Group (MANDATORY)		
2623 imGeneral OBJECT IDENTIFIER ::= { jobmonMIBObjects 1 } 2624 jmGeneralTable OBJECT-TYPE 2627 SYNTAX SEQUENCE OF JmGeneralEntry 2628 MAX-ACCESS not-accessible 2629 STATUS 2630 DESCRIPTION 2631 "The jmGeneralTable consists of information of a general nature that are per-job-set, but are 2632 mot per-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." 2633 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2634 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 imGeneralEntry OBJECT-TYPE 2636 SYNTAX JmGeneralEntry 2637 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX JmGeneralEntry 2640 STATUS current 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2644 imGeneralIobSetIndex } 2645 inDEX { jmGeneralIobSetIndex } 2646 := [jinGeneralIobSetIndex Kiteleolindex Kiteleolindex Kiteleolindex Kiteleolindex Kiteleolindex Kiteleolindex Kiteleolindex Kiteleolindex Kiteleolindex Kit	2621			
 jmGeneral OBJECT IDENTIFIER ::= { jobmonMIBObjects 1 } jmGeneralTable OBJECT-TYPE SYNTAX SEQUENCE OF JmGeneralEntry MAX-ACCESS not-accessible STATUS current DESCRIPTION The jmGeneralTable consists of information of a general nature that are per-job-set, but are not per-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." REFERENCE imGeneralEntry OBJECT-TYPE SYNTAX JmGeneralEntry MAX-ACCESS not-accessible SYNTAX JmGeneralEntry MAX-ACCESS not-accessible Tinformation about a job set (queue). imformation about a job set (queue).<!--</td--><td>2622</td><td> The jmGeneralGroup consists entirely of the jmGenera</td><td>lTable.</td>	2622	The jmGeneralGroup consists entirely of the jmGenera	lTable.	
2625 jmGeneralTable OBJECT-TYPE 2626 jmGeneralTable OBJECT-TYPE 2627 SYNTAX SEQUENCE OF JmGeneralEntry 2628 MAX-ACCESS not-accessible 2629 STATUS current 2630 DESCRIPTION 2631 "The jmGeneralTable consists of information of a general nature that are per-job-set, but are 2632 more-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." 2633 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 ::= { jmGeneralEntry 2636 SYNTAX JmGeneralEntry 2637 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX JmGeneralEntry 2639 MAX-ACCESS not-accessible 2640 STATUS current 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralJobSetIndex Integer32(02147483647), 2646 ::= { jmGeneralIobSetIndex Integer32(02147483647), 2651 jmGeneralNumberOIActiveJobIndex Integer32(02147483647),	2623			
 jmGeneralTable OBJECT-TYPE SYNTAX SEQUENCE OF JmGeneralEntry MAX-ACCESS not-accessible STATUS current DESCRIPTION The jmGeneralTable consists of information of a general nature that are per-job-set, but are not per-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." REFERENCE "The MANDATORY-GROUP macro specifies that this group is MANDATORY." ::= { jmGeneral 1 } SYNTAX JmGeneralEntry SYNTAX JmGeneralEntry MAX-ACCESS not-accessible SYNTAX JmGeneralEntry MAX-ACCESS not-accessible Integer32(0.2147483647), jmGeneralJobSetIndex OBJECT-TYPE jmGeneralJobSetIndex OBJECT-TYPE jmGeneralJobSetIndex OBJECT-TYPE SYNTAX Integer32(1.32767), jmGeneralJobSetIndex OBJECT-TYPE SYNTAX Integer32(1.32767), jmGeneralJobSetIndex OBJECT-TYPE SYNTAX Integer32(1.32767), jmGeneralJobSetIndex OBJECT-TYPE SYNTAX Integer32(1.32767), jmGeneralJobSetIndex OBJECT-TYPE DESCRIPTION MAX-ACCESS not-accessible SYNTAX Integer32(1.32767), DESCRIPTION DESCRIPTION DESCRIPTION JmGeneralJobSetIndex OBJECT-TYPE DESCRIPTION JmUTF8StringTC(SIZE(0.63)) JmGeneralJobSetIndex OBJECT-TYPE DESCRIPTION MAX-ACCESS Not-accessible SYNTAX Integer32(1.32767), DESCRIPTION 	2624	jmGeneral OBJECT IDENTIFIER ::= { jobmonMIBObject	ts 1 }	
2627 SYNTAX SEQUENCE OF JmGeneralEntry 2628 MAX-ACCESS not-accessible 2629 STATUS current 2630 DESCRIPTION 2631 "The jmGeneralTable consists of information of a general nature that are per-job-set, but are 2631 "The jmGeneralTable consists of information of a general nature that are per-job-set, but are 2634 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 ::= { jmGeneral 1 } 2636 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 ::= { jmGeneral 1 } 2636 "InfomentalEntry 2637 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX JmGeneralEntry 2640 STATUS current 2641 DESCRIPTION "Information about a job set (queue). 2642 "Information about a job set (queue). "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2000000000000000000000000000000000000	2625			
2628 MAX-ACCESS not-accessible 2629 STATUS current 2630 DESCRIPTION 2631 "The imGeneralTable consists of information of a general nature that are per-job-set, but are <i>not</i> per-job. See Section 2 entitled "Terminology and Job Model' for the definition of a job set." 2631 REFERENCE 2633 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 ::= { jmGeneral 1 } 2636 SYNTAX JmGeneralEntry 2637 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX JmGeneralEntry 2639 MAX-ACCESS not-accessible 2640 STATUS current 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralTable 1 } 2646 ::= { jmGeneralTable 1 } 2647 jmGeneralIobSetIndex Integer32(02147483647), 2650 jmGeneralNumberOfActiveJobs Integer32(02147483647), 2651 jmGeneralIobPersistence Integer32(152147483647), 2653	2626	jmGeneralTable OBJECT-TYPE		
2629 STATUS current 2630 DESCRIPTION 2631 "The jmGeneralTable consists of information of a general nature that are per-job-set, but are 2631 "The jmGeneralTable consists of information of a general nature that are per-job-set, but are 2632 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 ::= { jmGeneral 1 } 2636 SYNTAX 2637 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX 2640 STATUS 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralJobSetIndex } 2646 ::= { jmGeneralTable 1 } 2647 jmGeneralIobSetIndex Integer32(02147483647), 2650 jmGeneralNumberOfActiveJobs Integer32(02147483647), 2651 jmGeneralJobSetIndex Integer32(152147483647), 2652 jmGeneralJobPersistence Integer32(152147483647), 2653 jmGeneralJobSetIndex OBJECT-TYPE 2654 jmGeneralJobSetIndex OBJECT-TYPE		SYNTAX SEQUENCE OF JmGeneralEntry		
2630 DESCRIPTION 2631 "The jmGeneralTable consists of information of a general nature that are per-job-set, but are not per-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." 2633 REFERENCE 2634 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 ::= { jmGeneral 1 } 2636 ginGeneralEntry OBJECT-TYPE 2637 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX 2640 STATUS 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralJobSetIndex } 2646 ::= { jmGeneralJobSetIndex } 2647 ::= { jmGeneralJobSetIndex } 2648 ImGeneralDobSetIndex Integer32(02147483647), 2650 jmGeneralNumberOfActiveJobs Integer32(02147483647), 2651 jmGeneralJobPersistence Integer32(152147483647), 2653 jmGeneralJobSetIndex OBJECT-TYPE JmUTF8StringTC(SIZE(063)) 2654 jmGeneralJobSetIndex OBJECT-TYPE		MAX-ACCESS not-accessible		
2631 "The jmGeneralTable consists of information of a general nature that are per-job-set, but are 2632 nor per-job. See Section 2 entitled Terminology and Job Model' for the definition of a job set." 2633 REFERENCE 2634 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 :::= { jmGeneral 1 } 2636 ::::= { jmGeneral 1 } 2637 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX JmGeneralEntry 2639 MAX-ACCESS not-accessible 2640 STATUS current 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralJobSetIndex } 2646 ::= { jmGeneralTable 1 } 2647 jmGeneralTable 1 } 2648 JmGeneralNumberOfActiveJobs Integer32(02147483647), 2650 jmGeneralNumberOfActiveJobIndex Integer32(02147483647), 2651 jmGeneralActiveJobIndex Integer32(02147483647), 2652 jmGeneralJobSetIndex OBJECT-TYPE jmGeneralAttributePersist		STATUS current		
2632 not per-job. See Section 2 entitled "Terminology and Job Model' for the definition of a job set." 2634 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 ::= { jmGeneral 1 } 2636 imGeneralEntry OBJECT-TYPE 2637 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX JmGeneralEntry 2639 MAX-ACCESS not-accessible 2640 STATUS current 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralIobSetIndex } 2646 ::= { jmGeneralIobSetIndex Integer32(02147483647), 2651 jmGeneralNumberOfActiveJobs Integer32(02147483647), 2652 jmGeneralNumberOfActiveJobIndex Integer32(02147483647), 2653 jmGeneralJobSetIndex Integer32(02147483647), 2653 jmGeneralJobSetIndex Integer32(02147483647), 2654 jmGeneralJobSetIndex Integer32(02147483647), 2655 jmGeneralJobSetIndex OBJECT-TYPE 2656 jmGeneralJobSetIndex OBJECT-TYPE	2630	DESCRIPTION		
2633 REFERENCÉ The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2634 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 :::= { jmGeneral 1 } 2636 imGeneralEntry OBJECT-TYPE 2638 SYNTAX 2639 MAX-ACCESS not-accessible 2640 STATUS 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralJobSetIndex } 2646 ::= { jmGeneralTable 1 } 2647 JmGeneralInbestIndex X 2648 JmGeneralNumberOfActiveJobs 2649 jmGeneralNumberOfActiveJobs 2641 Integer32(02147483647), 2652 jmGeneralNotpersistence 2653 jmGeneralJobPersistence 2654 jmGeneralJobSetIndex 2655 jmGeneralJobSetIndex OBJECT-TYPE 2656 jmGeneralJobSetIndex OBJECT-TYPE 2656 jmGeneralJobSetIndex OBJECT-TYPE 2657 jmGeneralJobSetInd	2631	"The jmGeneralTable consists of information	of a general nature that are per-job-set, but are	
2634 "The MANDATORY-GROUP macro specifies that this group is MANDATORY." 2635 ::= { jmGeneral 1 } 2636 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX JmGeneralEntry 2639 MAX-ACCESS not-accessible 2640 STATUS current 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 An entry SHALL exist in this table for each job set." 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralJobSetIndex } 2646 ::= { jmGeneralTable 1 } 2647 ::= { jmGeneralTable 1 } 2648 JmGeneralIobSetIndex Integer32(02147483647), 2650 jmGeneralNumberOfActiveJobs Integer32(02147483647), 2651 jmGeneralDobSetIndex Integer32(02147483647), 2652 jmGeneralJobPersistence Integer32(152147483647), 2653 jmGeneralJobPersistence Integer32(152147483647), 2655 jmGeneralJobSetIndex OBJECT-TYPE 2656 jmGeneralJobSetIndex OBJECT-TYPE 2658 jmGeneralJobSetIndex OBJECT-TYPE 2659 SYNTAX Integer32(132767) 2660 MAX-ACEESS not	2632	not per-job. See Section 2 entitled 'Terminolog	y and Job Model' for the definition of a job set."	
2635 ::= { jmGeneral 1 } 2636 2637 jmGeneralEntry OBJECT-TYPE 2638 SYNTAX JmGeneralEntry 2639 MAX-ACCESS not-accessible 2640 STATUS current 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralJobSetIndex } 2646 ::= { jmGeneralJobSetIndex } 2647 2648 JmGeneralLobSetIndex Integer32(132767), 2650 jmGeneralJobSetIndex Integer32(02147483647), 2651 jmGeneralNumberOfActiveJobs Integer32(02147483647), 2652 jmGeneralNowestActiveJobIndex Integer32(02147483647), 2653 jmGeneralNowestActiveJobIndex Integer32(12147483647), 2654 jmGeneralNowestActiveJobIndex Integer32(152147483647), 2655 jmGeneralNowestActiveJobIndex Integer32(152147483647), 2654 jmGeneralJobSetIndex OBJECT-TYPE 2659 SYNTAX Integer32(132767) 2660 MAX-ACCESS not-accessible 2661 STATUS current 2662 DESCRIPTION	2633	REFERENCE		
2636 IN TAX JmGeneralEntry 2638 SYNTAX JmGeneralEntry 2639 MAX-ACCESS not-accessible 2640 STATUS current 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralJobSetIndex } 2646 ::= { jmGeneralJobSetIndex Integer32(132767), 2650 jmGeneralJobBetIndex Integer32(02147483647), 2651 jmGeneralJobBetIndex Integer32(02147483647), 2652 jmGeneralNumberOfActiveJobs Integer32(02147483647), 2653 jmGeneralNewestActiveJobIndex Integer32(12147483647), 2654 jmGeneralNewestActiveJobIndex Integer32(12147483647), 2655 jmGeneralNewestActiveJobIndex Integer32(12147483647), 2654 jmGeneralNewestActiveJobIndex Integer32(12147483647), 2655 jmGeneralJobSetIndex OBJECT-TYPE 2659 SYNTAX Integer32(132767) 2660 MAX-ACCESS not-accessible 2661 STATUS current 2662 DESCRIPTION	2634	"The MANDATORY-GROUP macro specifies	that this group is MANDATORY."	
 jmGeneralEntry OBJECT-TYPE SYNTAX JmGeneralEntry MAX-ACCESS not-accessible STATUS current DESCRIPTION Information about a job set (queue). an entry SHALL exist in this table for each job set." INDEX { jmGeneralJobSetIndex } ::= { jmGeneralTable 1 } ::= { jmGeneralTable 1 } jmGeneralIobSetIndex Integer32(132767), jmGeneralIobSetIndex Integer32(02147483647), jmGeneralIobPersistence Integer32(12147483647), jmGeneralJobPersistence Integer32(1.52147483647), jmGeneralJobSetIndex Integer32(1.52147483647), jmGeneralJobPersistence Integer32(1.52147483647), jmGeneralJobSetIndex jmGeneralJobSetIndex jmGeneralJobSetIndex integer32(1.52147483647), jmGeneralJobSetIndex jmGeneralJobSetIndex jmGeneralJobSetIndex jmGeneralJobSetIndex jmGeneralJobPersistence jmGeneralJobSetIndex jmG	2635	::= { jmGeneral 1 }		
2638 SYNTAX ImGeneralEntry 2639 MAX-ACCESS not-accessible 2640 STATUS current 2641 DESCRIPTION 2642 "Information about a job set (queue). 2643 2644 An entry SHALL exist in this table for each job set." 2645 INDEX { jmGeneralJobSetIndex } 2646 ::= { jmGeneralTable 1 } 2647 2648 JmGeneralEntry ::= SEQUENCE { 2649 jmGeneralJobSetIndex Integer32(132767), 2650 jmGeneralOldestActiveJobIndex Integer32(02147483647), 2651 jmGeneralOldestActiveJobIndex Integer32(02147483647), 2652 jmGeneralJobPersistence Integer32(152147483647), 2653 jmGeneralJobPersistence Integer32(152147483647), 2654 jmGeneralJobSetName JmUTF8StringTC(SIZE(063)) 2656 } 2658 jmGeneralJobSetIndex OBJECT-TYPE 2659 SYNTAX Integer32(132767) 2660 MAX-ACCESS not-accessible 2661 STATUS current 2662 DESCRIPTION				
2639MAX-ACCESS not-accessible2640STATUS current2641DESCRIPTION2642"Information about a job set (queue).2643An entry SHALL exist in this table for each job set."2644An entry SHALL exist in this table for each job set."2645INDEX { jmGeneralJobSetIndex }2646::= { jmGeneralTable 1 }2647				
 STATUS current DESCRIPTION Information about a job set (queue). "Information about a job set (queue). An entry SHALL exist in this table for each job set." INDEX { jmGeneralJobSetIndex } INDEX { jmGeneralJobSetIndex } Integer32(132767), jmGeneralNumberOfActiveJobs Integer32(02147483647), jmGeneralNewestActiveJobIndex Integer32(152147483647), jmGeneralJobSetIndex Integer32(152147483647), jmGeneralJobPersistence Integer32(152147483647), jmGeneralJobSetIndex OBJECT-TYPE jmGeneralJobSetIndex OBJECT-TYPE SYNTAX Integer32(132767) MAX-ACCESS not-accessible STATUS current DESCRIPTION 				
2641DESCRIPTION2642"Information about a job set (queue).2643An entry SHALL exist in this table for each job set."2644An entry SHALL exist in this table for each job set."2645INDEX { jmGeneralJobSetIndex }2646::= { jmGeneralTable 1 }2647ImGeneralEntry ::= SEQUENCE {2648JmGeneralNumberOfActiveJobs2650jmGeneralOldestActiveJobIndex2651jmGeneralOldestActiveJobIndex2652jmGeneralJobPersistence2653jmGeneralJobPersistence2654jmGeneralJobSetIndex2655jmGeneralJobSetIndex2656JmGeneralJobSetIndex2657JmGeneralJobSetIndex2658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAX2650MAX-ACCESS not-accessible2661STATUS2662DESCRIPTION				
2642"Information about a job set (queue).2643An entry SHALL exist in this table for each job set."2644An entry SHALL exist in this table for each job set."2645INDEX { jmGeneralJobSetIndex }2646::= { jmGeneralTable 1 }2647				
2643An entry SHALL exist in this table for each job set."2644An entry SHALL exist in this table for each job set."2645INDEX { jmGeneralJobSetIndex }2646::= { jmGeneralTable 1 }264726482648JmGeneralEntry ::= SEQUENCE {2649jmGeneralJobSetIndex2650jmGeneralOldestActiveJobIndex2651jmGeneralOldestActiveJobIndex2652jmGeneralNewestActiveJobIndex2653jmGeneralJobPersistence2654Integer32(02147483647),2655jmGeneralJobPersistence2656JmGeneralJobSetName2657JmGeneralJobSetIndex OBJECT-TYPE2658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAX2660MAX-ACCESS not-accessible2661STATUS2662DESCRIPTION				
2644An entry SHALL exist in this table for each job set."2645INDEX { jmGeneralJobSetIndex }2646::= { jmGeneralTable 1 }264726482647JmGeneralEntry ::= SEQUENCE {2649jmGeneralJobSetIndex2650jmGeneralNumberOfActiveJobs2651jmGeneralNumberOfActiveJobs2652jmGeneralNewestActiveJobIndex2653jmGeneralNewestActiveJobIndex2654Integer32(02147483647),2655jmGeneralNewestActiveJobIndex2654Integer32(132767),2655jmGeneralAttributePersistence2656Integer32(152147483647),2657jmGeneralJobSetIndex OBJECT-TYPE2658jmGeneralJobSetIndex OBJECT-TYPE2660MAX-ACCESS not-accessible2661STATUS current2662DESCRIPTION		"Information about a job set (queue).		
2645INDEX { jmGeneralJobSetIndex }2646::= { jmGeneralTable 1 }264726482648JmGeneralEntry ::= SEQUENCE {2649jmGeneralJobSetIndex2650jmGeneralNumberOfActiveJobs2651jmGeneralOldestActiveJobIndex2652jmGeneralNewestActiveJobIndex2653jmGeneralJobPersistence2654jmGeneralJobPersistence2655jmGeneralJobSetIndex2656}26572658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAX2659SYNTAX2660MAX-ACCESS not-accessible2661STATUS262DESCRIPTION				
2646 ::= { jmGeneralTable 1 } 2647 2648 JmGeneralEntry ::= SEQUENCE { 2649 jmGeneralJobSetIndex Integer32(132767), 2650 jmGeneralOldestActiveJobs Integer32(02147483647), 2651 jmGeneralNewestActiveJobIndex Integer32(02147483647), 2652 jmGeneralNewestActiveJobIndex Integer32(02147483647), 2653 jmGeneralJobPersistence Integer32(152147483647), 2654 jmGeneralAttributePersistence Integer32(152147483647), 2655 jmGeneralJobSetName JmUTF8StringTC(SIZE(063))) 2656 } 2657 2657 jmGeneralJobSetIndex OBJECT-TYPE 2658 jmGeneralJobSetIndex OBJECT-TYPE 2659 2659 SYNTAX Integer32(132767) 2660 MAX-ACCESS not-accessible 2661 2661 STATUS current 2662 DESCRIPTION JmStringTion			set."	
26472648JmGeneralEntry ::= SEQUENCE {2649jmGeneralJobSetIndexInteger32(132767),2650jmGeneralNumberOfActiveJobsInteger32(02147483647),2651jmGeneralOldestActiveJobIndexInteger32(02147483647),2652jmGeneralNewestActiveJobIndexInteger32(02147483647),2653jmGeneralJobPersistenceInteger32(152147483647),2654jmGeneralAttributePersistenceInteger32(152147483647),2655jmGeneralJobSetNameJmUTF8StringTC(SIZE(063))2656}26572658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAXInteger32(132767)2660MAX-ACCESS not-accessible2661STATUScurrent2662DESCRIPTION				
2648JmGeneralEntry ::= SEQUENCE {2649jmGeneralJobSetIndexInteger32(132767),2650jmGeneralNumberOfActiveJobsInteger32(02147483647),2651jmGeneralOldestActiveJobIndexInteger32(02147483647),2652jmGeneralNewestActiveJobIndexInteger32(02147483647),2653jmGeneralJobPersistenceInteger32(152147483647),2654jmGeneralAttributePersistenceInteger32(152147483647),2655jmGeneralJobSetNameJmUTF8StringTC(SIZE(063))2656}		::= { jmGeneralTable 1 }		
2649jmGeneralJobSetIndexInteger32(132767),2650jmGeneralNumberOfActiveJobsInteger32(02147483647),2651jmGeneralOldestActiveJobIndexInteger32(02147483647),2652jmGeneralNewestActiveJobIndexInteger32(02147483647),2653jmGeneralJobPersistenceInteger32(152147483647),2654jmGeneralAttributePersistenceInteger32(152147483647),2655jmGeneralJobSetNameJmUTF8StringTC(SIZE(063))2656}26572658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAXInteger32(132767)2660MAX-ACCESS not-accessible2661STATUScurrent262DESCRIPTION				
2650jmGeneralNumberOfActiveJobsInteger32(02147483647),2651jmGeneralOldestActiveJobIndexInteger32(02147483647),2652jmGeneralNewestActiveJobIndexInteger32(02147483647),2653jmGeneralJobPersistenceInteger32(152147483647),2654jmGeneralAttributePersistenceInteger32(152147483647),2655jmGeneralJobSetNameJmUTF8StringTC(SIZE(063))2656}				
2651jmGeneralOldestActiveJobIndexInteger32(02147483647),2652jmGeneralNewestActiveJobIndexInteger32(02147483647),2653jmGeneralJobPersistenceInteger32(152147483647),2654jmGeneralAttributePersistenceInteger32(152147483647),2655jmGeneralJobSetNameJmUTF8StringTC(SIZE(063))2656}26572658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAXInteger32(132767)2660MAX-ACCESS not-accessible2661STATUS262DESCRIPTION				
2652jmGeneralNewestActiveJobIndexInteger32(02147483647),2653jmGeneralJobPersistenceInteger32(152147483647),2654jmGeneralAttributePersistenceInteger32(152147483647),2655jmGeneralJobSetNameJmUTF8StringTC(SIZE(063))2656}26572658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAXInteger32(132767)2660MAX-ACCESS not-accessible2661STATUS262DESCRIPTION				
2653jmGeneralJobPersistenceInteger32(152147483647),2654jmGeneralAttributePersistenceInteger32(152147483647),2655jmGeneralJobSetNameJmUTF8StringTC(SIZE(063))2656}26572658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAXInteger32(132767)2660MAX-ACCESS not-accessible2651STATUS2662DESCRIPTION			Integer 32(02147483647),	
2654jmGeneralAttributePersistenceInteger32(152147483647),2655jmGeneralJobSetNameJmUTF8StringTC(SIZE(063))2656}26572658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAX2659SYNTAX2660MAX-ACCESS not-accessible2661STATUS262DESCRIPTION				
2655jmGeneralJobSetNameJmUTF8StringTC(SIZE(063))2656}26572658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAX2659SYNTAX2660MAX-ACCESS not-accessible2661STATUS2662DESCRIPTION				
 2656 } 2657 2658 jmGeneralJobSetIndex OBJECT-TYPE 2659 SYNTAX Integer32(132767) 2660 MAX-ACCESS not-accessible 2661 STATUS current 2662 DESCRIPTION 				
26572658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAX Integer32(132767)2660MAX-ACCESS not-accessible2661STATUS current2662DESCRIPTION			JmU1F88tring1C(SIZE(003))	
2658jmGeneralJobSetIndex OBJECT-TYPE2659SYNTAX Integer32(132767)2660MAX-ACCESS not-accessible2661STATUS current2662DESCRIPTION		}		
2659SYNTAXInteger32(132767)2660MAX-ACCESS not-accessible2661STATUS2662DESCRIPTION		mCanaval JahSatInday OD JECT TVDE		
2660MAX-ACCESS not-accessible2661STATUS current2662DESCRIPTION				
2661STATUS current2662DESCRIPTION				
2662 DESCRIPTION				
2005 If and a value for each job set in ans with, the juisou faste and juistic faste faste faste			he im.IohTable and imAttributeTable tables	
2664 have this same index as their primary index.			in juicourable and juicturibute rable tables	
2665		have and baile index as then printing index.		

2666 2667	The value(s) of the jmGeneralJobSetIndex SHALL be persistent across power cycles, so that clients that have retained jmGeneralJobSetIndex values will access the same job sets upon
2668 2669	subsequent power-up.
2670 2671	An implementation that has only one job set, such as a printer with a single queue, SHALL hard code this object with the value 1."
2672	REFERENCE
2673	"See Section 2 entitled 'Terminology and Job Model' for the definition of a job set.
2674	Corresponds to the first index in jmJobTable and jmAttributeTable ."
2675	::= { jmGeneralEntry 1 }
2676	
2677	jmGeneralNumberOfActiveJobs OBJECT-TYPE
2678	SYNTAX Integer32(02147483647)
2679	MAX-ACCESS read-only
2680	STATUS current
2681	DESCRIPTION
2682	"The current number of 'active' jobs in the jmJobIDTable , jmJobTable , and
2683	jmAttributeTable , i.e., the total number of jobs that are in the pending , processing , or
2684	processingStopped states. See the JmJobStateTC textual-convention for the exact
2685	specification of the semantics of the job states."
2686	::= { jmGeneralEntry 2 }
2687	
2688	jmGeneralOldestActiveJobIndex OBJECT-TYPE
2689	SYNTAX Integer32 (02147483647)
2690	MAX-ACCESS read-only
2691	STATUS current
2692	DESCRIPTION
2693	"The jmJobIndex of the oldest job that is still in one of the 'active' states (pending , processing ,
2694	or processingStopped). In other words, the index of the 'active' job that has been in the job
2695	tables the longest.
2696	
2697	If there are no active jobs, the agent SHALL set the value of this object to 0."
2698	REFERENCE
2699	"See Section 3.2 entitled 'The Job Tables and the Oldest Active and Newest Active Indexes' for
2700	a description of the usage of this object."
2701	::= { jmGeneralEntry 3 }
2702	
2703	jmGeneralNewestActiveJobIndex OBJECT-TYPE
2704	SYNTAX Integer32 (02147483647)
2705	MAX-ACCESS read-only
2706	STATUS current
2707	DESCRIPTION
2708	"The jmJobIndex of the newest job that is in one of the 'active' states (pending , processing , or
2709	processingStopped). In other words, the index of the 'active' job that has been most recently
2710	added to the job tables.
2711	
2712	When all jobs become 'inactive', i.e., enter the pendingHeld , completed , canceled , or aborted
2713	states, the agent SHALL set the value of this object to 0 ."
2714	REFERENCE

Aug 8, 1997

2715 "See Section 3.2 entitled 'The Job Tables and the Oldest Active and Newest Active Indexes' for 2716 a description of the usage of this object." 2717 ::= { jmGeneralEntry 4 } 2718 2719 jmGeneralJobPersistence OBJECT-TYPE 2720 SYNTAX Integer32(15..2147483647) 2721 UNITS "seconds" 2722 MAX-ACCESS read-only 2723 STATUS current 2724 DESCRIPTION 2725 "The minimum time in seconds for this instance of the Job Set that an entry SHALL remain in 2726 the **jmJobIDTable** and **jmJobTable** after **processing** has *completed*, i.e., the minimum time in 2727 seconds starting when the job enters the **completed**, **canceled**, **or aborted** state. 2728 2729 Depending on implementation, the value of this object MAY be either: (1) set by the system 2730 administrator by means outside this specification or (2) fixed by the implementation. 2731 This value SHALL be equal to or greater than the value of jmGeneralAttributePersistence. 2732 2733 This value SHOULD be at least 60 which gives a monitoring application one minute in which to 2734 poll for job data." 2735 DEFVAL { 60 } -- one minute 2736 ::= { jmGeneralEntry 5 } 2737 2738 jmGeneralAttributePersistence OBJECT-TYPE 2739 SYNTAX Integer32(15..2147483647) "seconds" 2740 UNITS MAX-ACCESS read-only 2741 current 2742 STATUS 2743 DESCRIPTION 2744 "The minimum time in seconds for this instance of the Job Set that an entry SHALL remain in the **jmAttributeTable** after **processing** has *completed*, i.e., the time in seconds starting when 2745 2746 the job enters the **completed**, **canceled**, or **aborted** state. 2747 2748 Depending on implementation, the value of this object MAY be either (1) set by the system 2749 administrator by means outside this specification or MAY be (2) fixed by the implementation. 2750 This value SHOULD be at least 60 which gives a monitoring application one minute in which to 2751 poll for job data." 2752 DEFVAL 2753 { 60 } -- one minute 2754 ::= { jmGeneralEntry 6 } 2755 2756 jmGeneralJobSetName OBJECT-TYPE 2757 JmUTF8StringTC(SIZE(0..63)) SYNTAX 2758 MAX-ACCESS read-only 2759 **STATUS** current 2760 DESCRIPTION 2761 "The human readable name of this job set assigned by the system administrator (by means outside of this MIB). Typically, this name SHOULD be the name of the job queue. If a server 2762 2763 or device has only a single job set, this object can be the administratively assigned name of the

2764 2765	server or device itself. This name does not need to be unique, though each job set in a single Job Monitoring MIB SHOULD have distinct names.		
2766			
2767	NOTE - The purpose of th	s object is to help the user of the job monitoring application	
2768		l job sets in implementations that support more than one job set."	
		1 job sets in implementations that support more than one job set.	
2769	REFERENCE		
2770		nce macro for the minimum maximum length required for	
2771	conformance."		
2772	::= { jmGeneralEntry 7 }		
2773			
2774			
2775			
2776			
2777			
2778	The Job ID Group (MANDATORY		
2779		1	
	The im LabIDC norm consists entirely	v of the im IchIDT able	
2780	The jmJobIDGroup consists entire	y of the JHJODID Lable.	
2781			
2782	jmJobID OBJECT IDENTIFIER ::= {	jobmonMIBObjects 2 }	
2783			
2784	jmJobIDTable OBJECT-TYPE		
2785	SYNTAX SEQUENCE OF J	mJobIDEntry	
2786	MAX-ACCESS not-accessible		
2787	STATUS current		
2788	DESCRIPTION		
2789		des a correspondence map (1) between the job submission ID that a	
2790		and (2) the jmGeneralJobSetIndex and jmJobIndex that the Job	
2791		igned to the job and that are used to access the job in all of the other	
2792		nitoring application already knows the jmGeneralJobSetIndex and	
2793		it is querying, that application NEED NOT use the jmJobIDTable ."	
		it is querying, that application NEED NOT use the JIIJODID Table .	
2794	REFERENCE	NUD and an a first that this second is MANDATODY "	
2795		OUP macro specifies that this group is MANDATORY."	
2796	::= { jmJobID 1 }		
2797			
2798	jmJobIDEntry OBJECT-TYPE		
2799	SYNTAX JmJobIDEntry		
2800	MAX-ACCESS not-accessible		
2801	STATUS current		
2802	DESCRIPTION		
2803		JobSubmissionID to (2) the jmGeneralJobSetIndex and	
2804	jmJobIndex.		
2805			
2806	An entry SHALL exist in t	his table for each job currently known to the agent for all job sets and	
2807		L appear in one and only one job set."	
2808	INDEX { jmJobSubmissionID		
2809	$::= \{ jmJobIDTable 1 \}$,	
2810			
2811	JmJobIDEntry ::= SEQUENCE {		
2812	jmJobSubmissionID	OCTET STRING(SIZE(48)) ,	
	J		

2813 2814 2815	jmJobIDJobSetIndex jmJobIDJobIndex	Integer32(132767), Integer32(12147483647)
	}	
2816 2817	jmJobSubmissionID OBJECT-TYPI	
2817	SYNTAX OCTET STRING	
2818	MAX-ACCESS not-accessible	f(SIZE(40))
281)	STATUS current	
2820	DESCRIPTION	
2822		fixed-length string ID which identifies the job within a particular
2823		There are multiple formats for the jmJobSubmissionID . Each
2824		y identified. See the JmJobSubmissionIDTypeTC textual convention.
2825		egistered using the procedures of a type 2 enum. See section 3.6.3
2826	entitled: 'IANA Registration	on of Job Submission Id Formats'.
2827	C	
2828	If the requester (client or s	erver) does not supply a job submission ID in the job submission
2829		t (server or device) SHALL assign a job submission ID using any of
2830		have been reserved to agents and adding the final 8 octets to
2831	distinguish the ID from ot	hers submitted from the same requester.
2832	-	-
2833		n, whether in the client or running separately, MAY use the job
2834		tify which jmJobIndex was assigned by the agent, i.e., in which row
2835	the job information is in the	e other tables.
2836		
2837		ed so that a management application can use a shortened GetNext
2838		SNMPv2) in order to get the next submission ID, disregarding the
2839		er to access jobs independent of the trailing identifier part, e.g., to get
2840		ticular jmJobOwner or submitted from a particular MAC address."
2841	REFERENCE	
2842		onIDTypeTC textual convention.
2843		ort of the Job Submission ID in Job Submission Protocols."
2844 2845	::= { jmJobIDEntry 1 }	
2845 2846	jmJobIDJobSetIndex OBJECT-TYF	
2840 2847	SYNTAX Integer32(1327	
2848	MAX-ACCESS read-only	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
2849	STATUS current	
2850	DESCRIPTION	
2851		value of the jmGeneralJobSetIndex for the job with the
2852		i.e., the job set index of the job set in which the job was placed
2853		accepted the job. This 16-bit value in combination with the
2854		permits the management application to access the other tables to
2855	obtain the job-specific obj	
2856	REFERENCE	5
2857		dex in the jmGeneralTable."
2858	::= { jmJobIDEntry 2 }	~
2859	- · · ·	
2860	jmJobIDJobIndex OBJECT-TYPE	
2861	SYNTAX Integer32(1214'	/483647)

2864	DESCRIPTION
2865	"This object contains the value of the jmJobIndex for the job with the jmJobSubmissionID
2866	value, i.e., the job index for the job when the server or device accepted the job. This value, in
2867	combination with the jmJobIDJobSetIndex value, permits the management application to
2868	access the other tables to obtain the job-specific objects for this job."
2869	REFERENCE
2870	"See jmJobIndex in the jmJobTable."
2871	::= { jmJobIDEntry 3 }
2872	
2873	
2874	
2875	
2876	The Job Group (MANDATORY)
2877	
2878	The jmJobGroup consists entirely of the jmJobTable .
2879	
2880	jmJob OBJECT IDENTIFIER ::= { jobmonMIBObjects 3 }
2881	
2882	jmJobTable OBJECT-TYPE
2883	SYNTAX SEQUENCE OF JmJobEntry
2884	MAX-ACCESS not-accessible
2885	STATUS current
2886	DESCRIPTION
2887	"The jmJobTable consists of basic job state and status information for each job in a job set that
2888	(1) monitoring applications need to be able to access in a single SNMP Get operation, (2) that
2889	have a single value per job, and (3) that SHALL always be implemented." REFERENCE
2890	
2891 2892	"The MANDATORY-GROUP macro specifies that this group is MANDATORY."
2892	::= { jmJob 1 }
2893	im JohEntry OBJECT TVDE
2895	jmJobEntry OBJECT-TYPE SYNTAX JmJobEntry
2895	MAX-ACCESS not-accessible
2897	STATUS current
2898	DESCRIPTION
2899	"Basic per-job state and status information.
2900	Dusie per job state and status information.
2901	An entry SHALL exist in this table for each job, no matter what the state of the job is. Each job
2902	SHALL appear in one and only one job set."
2903	REFERENCE
2904	"See Section 3.2 entitled 'The Job Tables'."
2905	INDEX { jmGeneralJobSetIndex, jmJobIndex }
2906	$::= \{ \text{ jmJobTable 1 } \}$
2907	
	JmJobEntry ::= SEQUENCE {
	jmJobIndex Integer32(12147483647),
2910	jmJobState JmJobStateTC,
2908 2909	

2911 2912 2913 2914 2915 2916 2917 2918 2919	jmJobStateReasons1 jmNumberOfInterveningJobs jmJobKOctetsRequested jmJobKOctetsProcessed jmJobImpressionsRequested jmJobImpressionsCompleted jmJobOwner }	JmJobStateReasons1TC, Integer32(-22147483647), Integer32(-22147483647), Integer32(-22147483647), Integer32(-22147483647), Integer32(-22147483647), JmJobStringTC(SIZE(063))	
2919	jmJobIndex OBJECT-TYPE		
2921	SYNTAX Integer32(12147483647)		
2922	MAX-ACCESS not-accessible		
2923	STATUS current		
2924	DESCRIPTION		
2925	"The sequential, monatonically increasing identified	er index for the job generated by the server or	
2926	device when that server or device accepted the job		
2927	application to access the other tables to obtain the		
2928	REFERENCE	job specific fow charles.	
2929	"See Section 3.2 entitled 'The Job Tables and the	Oldest Active and Newest Active Indexes'.	
2930	See Section 3.4 entitled 'Job Identification'.		
2931	See also jmGeneralNewestActiveJobIndex for the largest value of jmJobIndex.		
2932	See JmJobSubmissionTypeTC for a limit on the size of this index if the agent represents it as		
2933	an 8-digit decimal number."		
2934	::= { jmJobEntry 1 }		
2935			
2936	jmJobState OBJECT-TYPE		
2937	SYNTAX JmJobStateTC		
2938	MAX-ACCESS read-only		
2939	STATUS current		
2940	DESCRIPTION		
2941	"The current state of the job (pending, processing, completed, etc.). Agents SHALL		
2942	implement only those states which are appropriate for the particular implementation. However,		
2943	management applications SHALL be prepared to	receive all the standard job states.	
2944			
2945	The final value for this object SHALL be one of: c		
2946	minimum length of time that the agent SHALL ma		
2947	canceled, or aborted state before removing the jo	b data from the jmJobID lable and	
2948	jmJobTable is specified by the value of the jmGe	eneralJobPersistence object."	
2949	::= { jmJobEntry 2 }		
2950	im Joh State Desgenal OD IECT TYDE		
2951	jmJobStateReasons1 OBJECT-TYPE		
2952 2953	SYNTAX JmJobStateReasons1TC		
2953 2954	MAX-ACCESS read-only		
2954 2955	STATUS current DESCRIPTION		
2933 2956	DESCRIPTION "Additional information about the job's current state, i.e., information that augments the value of		
2950 2957	the job's jmJobState object.	ic, i.e., information that augments the value of	
2957	me job's jingobstate object.		
2930			

2959 2960 2961 2962 2963	Implementation of any reason values is OPTIONAL, but an agent SHOULD return any reason information available These values MAY be used with any job state or states for which the reason makes sense. Furthermore, when implemented as with any MIB data, the agent SHALL return these values when the reason applies and SHALL NOT return them when the reason no longer applies whether the value of the job's jmJobState object changed or not. When the
2964 2965 2966	agent cannot provide a reason for the current state of the job, the agent SHALL set the value of the jmJobStateReasons1 object and jobStateReasonsN attributes to 0 ." REFERENCE
2900 2967	"The jobStateReasonsN ($N=24$) attributes provide further additional information about the
2967	job's current state."
2968	$::= \{ jmJobEntry 3 \}$
2970	= { JIIDODENTLY 5 }
2971	jmNumberOfInterveningJobs OBJECT-TYPE
2972	SYNTAX Integer32(-22147483647)
2973	MAX-ACCESS read-only
2974	STATUS current
2975	DESCRIPTION
2976	"The number of jobs that are expected to complete being processed before this job has
2977	completed being processed according to the implementation's queuing algorithm if no other jobs
2978	were to be submitted. In other words, this value is the job's queue position. The agent SHALL
2979	return a value of 0 for this attribute when the job is the next job to complete processing (or has
2980	completed processing)."
2981	::= { jmJobEntry 4 }
2982	
2983	jmJobKOctetsRequested OBJECT-TYPE
2984	SYNTAX Integer32(-22147483647)
2985	MAX-ACCESS read-only
2986	STATUS current
2987	DESCRIPTION
2988	"The total size in K (1024) octets of the document(s) being requested to be processed in the job.
2989	The agent SHALL round the actual number of octets up to the next highest K. Thus 0 octets
2990	SHALL be represented as '0', 1-1024 octets SHALL be represented as '1', 1025-2048 SHALL
2991	be represented as '2', etc.
2992	In computing this value, the company/device SULATE, and include the multiplicative factors
2993 2994	In computing this value, the server/device SHALL <i>not</i> include the multiplicative factors contributed by (1) the number of document copies, and (2) the number of job copies,
2994 2995	independent of whether the device can process multiple copies of the job or document without
2995	making multiple passes over the job or document data and independent of whether the output is
2997	collated or not. Thus the server/device computation is independent of the implementation."
2998	::= { jmJobEntry 5 }
2999	(JillooLindy 5 J
3000	jmJobKOctetsProcessed OBJECT-TYPE
3001	SYNTAX Integer32(-22147483647)
3002	MAX-ACCESS read-only
3003	STATUS current
3004	DESCRIPTION
3005	"The current number of octets processed by the server or device measured in units of K (1024)
3006	octets. The agent SHALL round the actual number of octets processed up to the next higher K.
3007	Thus 0 octets SHALL be represented as '0', 1-1024 octets SHALL be represented as '1', 1025-

3008 2048 octets SHALL be '2', etc. For printing devices, this value is the number interpreted by the 3009 page description language interpreter rather than what has been marked on media. 3010 3011 For implementations where multiple copies are produced by the interpreter with only a single pass over the data, the final value SHALL be equal to the value of the 3012 3013 jmJobKOctetsRequested object. For implementations where multiple copies are produced by the interpreter by processing the data for each copy, the final value SHALL be a multiple of the 3014 3015 value of the **jmJobKOctetsRequested** object. 3016 NOTE - See the **impressionsCompletedCurrentCopy** and **pagesCompletedCurrentCopy** 3017 3018 attributes for attributes that are reset on each document copy. 3019 3020 NOTE - The **jmJobKOctetsProcessed** object can be used with the **jmJobKOctetsRequested** 3021 object to provide an indication of the relative progress of the job, provided that the 3022 multiplicative factor is taken into account for some implementations of multiple copies." 3023 ::= { jmJobEntry 6 } 3024 3025 jmJobImpressionsRequested OBJECT-TYPE 3026 SYNTAX Integer32(-2..2147483647) 3027 MAX-ACCESS read-only 3028 STATUS current 3029 DESCRIPTION 3030 "The total size in number of impressions of the document(s) being requested by this job to 3031 produce. 3032 3033 In computing this value, the server/device SHALL *not* include the multiplicative factors contributed by (1) the number of document copies, and (2) the number of job copies, 3034 3035 independent of whether the device can process multiple copies of the job or document without 3036 making multiple passes over the job or document data and independent of whether the output is 3037 collated or not. Thus the server/device computation is independent of the implementation." 3038 $::= \{ \text{ jmJobEntry } 7 \}$ 3039 3040 jmJobImpressionsCompleted OBJECT-TYPE 3041 SYNTAX Integer32(-2..2147483647) 3042 MAX-ACCESS read-only 3043 STATUS current DESCRIPTION 3044 "The current number of impressions completed for this job so far. For printing devices, the 3045 3046 impressions completed includes interpreting, marking, and stacking the output. For other types 3047 of job services, the number of impressions completed includes the number of impressions 3048 processed. 3049 3050 For implementations where multiple copies are produced by the interpreter with only a single 3051 pass over the data, the final value SHALL be equal to the value of the 3052 **jmJobImpressionsRequested** object. For implementations where multiple copies are produced 3053 by the interpreter by processing the data for each copy, the final value SHALL be a multiple of 3054 the value of the **jmJobImpressionsRequested** object. 3055

3056	NOTE - See the impressionsCompletedCurrentCopy and pagesCompletedCurrentCopy
3057	attributes for attributes that are reset on each document copy.
3058	
3059	NOTE - The jmJobImpressionsCompleted object can be used with the
3060	jmJobImpressionsRequested object to provide an indication of the relative progress of the job,
3061	provided that the multiplicative factor is taken into account for some implementations of
3062	multiple copies."
3063	::= { jmJobEntry 8 }
3064	
3065	jmJobOwner OBJECT-TYPE
3066	SYNTAX JmJobStringTC(SIZE(063))
3067	MAX-ACCESS read-only
3068	STATUS current
3069	DESCRIPTION
3070	"The coded character set name of the user that submitted the job. The method of assigning this
3071	user name will be system and/or site specific but the method MUST insure that the name is
3072	unique to the network that is visible to the client and target device.
3073	
3074	This value SHOULD be the <i>authenticated</i> name of the user submitting the job."
3075	REFERENCE
3076	"See the OBJECT compliance macro for the minimum maximum length required for
3077	conformance."
3078	::= { jmJobEntry 9 }
3079	
3080	
3081	
3082	
3083	The Attribute Group (MANDATORY)
3084	
3085	The jmAttributeGroup consists entirely of the jmAttributeTable.
3086	
3087	Implementation of the two objects in this group is MANDATORY.
3088	See Section 3.1 entitled 'Conformance Considerations'.
3089	An agent SHALL implement any attribute if (1) the server or device
3090	supports the functionality represented by the attribute and (2) the
3091	information is available to the agent.
3092	
3093	jmAttribute OBJECT IDENTIFIER ::= { jobmonMIBObjects 4 }
3094	
3095	jmAttributeTable OBJECT-TYPE
3096	SYNTAX SEQUENCE OF JmAttributeEntry
3097	MAX-ACCESS not-accessible
3098	STATUS current
3099	DESCRIPTION
3100	"The jmAttributeTable SHALL contain attributes of the job and document(s) for each job in a
3101	job set. Instead of allocating distinct objects for each attribute, each attribute is represented as a
3102	separate row in the jmAttributeTable ."
3103	REFERENCE

3104 3105 3106 3107 3108	<pre>"The MANDATORY-GROUP macro specifies that this group is MANDATORY. An agent SHALL implement any attribute if (1) the server or device supports the functionality represented by the attribute and (2) the information is available to the agent. " ::= { jmAttribute 1 }</pre>			
3109	jmAttributeEntry OBJECT-TYPE			
3110	SYNTAX JmAttributeEntry			
3111	MAX-ACCESS not-accessible			
3112	STATUS current			
3113	DESCRIPTION			
3114	"Attributes representing information about the job and document(s) or resources required			
3115	consumed.			
3116				
3117	Each entry in the jmAttributeTable is a per-job entry with an extra index for each type of			
3118	attribute (jmAttributeTypeIndex) that a job can have and an additional index			
3119	(jmAttributeInstanceIndex) for those attributes that can have multiple instances per job.	The		
3120	jmAttributeTypeIndex object SHALL contain an enum type that indicates the type of att			
3121	(see the JmAttributeTypeTC textual-convention). The value of the attribute SHALL be			
3122	represented in either the jmAttributeValueAsInteger or jmAttributeValueAsOctets obj	ects,		
3123	and/or both, as specified in the JmAttributeTypeTC textual-convention.			
3124				
3125	The agent SHALL create rows in the jmAttributeTable as the server or device is able to			
3126	discover the attributes either from the job submission protocol itself or from the document			
3127	As the documents are interpreted, the interpreter MAY discover additional attributes and s			
3128	agent adds additional rows to this table. As the attributes that represent resources are actually			
3129	consumed, the usage counter contained in the jmAttributeValueAsInteger object is			
3130	incremented according to the units indicated in the description of the JmAttributeTypeTC			
3131	enum.			
3132				
3133	The agent SHALL maintain each row in the jmJobTable for at least the minimum time after the second	er a		
3134	job completes as specified by the jmGeneralAttributePersistence object.			
3135	Zere on more entries SUALL exist in this table for each ich in a ich act "			
3136	Zero or more entries SHALL exist in this table for each job in a job set."			
3137 3138	REFERENCE "See Section 3.3 entitled 'The Attribute Mechanism' for a description of the jmAttributeT	abla "		
3138	INDEX { jmGeneralJobSetIndex, jmJobIndex, jmAttributeTypeIndex,	able.		
3139	jmAttributeInstanceIndex }			
3140	::= { jmAttributeTable 1 }			
3141				
3143	JmAttributeEntry ::= SEQUENCE {			
3144	jmAttributeTypeIndex JmAttributeTypeTC,			
3145	jmAttributeInstanceIndex Integer32(132767),			
3146	jmAttributeValueAsInteger Integer32(-22147483647),			
3147	jmAttributeValueAsOctets OCTET STRING(SIZE(063))			
3148	j			
3149				
3150	jmAttributeTypeIndex OBJECT-TYPE			
3151	SYNTAX JmAttributeTypeTC			
3152	MAX-ACCESS not-accessible			

3153	STATUS current
3154	DESCRIPTION
3155	"The type of attribute that this row entry represents.
3156	
3157	The type MAY identify information about the job or document(s) or MAY identify a resource
3158	required to process the job before the job start processing and/or consumed by the job as the job
3159	is processed.
3160	
3161	Examples of job attributes (i.e., apply to the job as a whole) that have only one instance per job
3162	include: jobCopiesRequested(90), documentCopiesRequested(92),
3163	jobCopiesCompleted(91), documentCopiesCompleted(93), while examples of job attributes
3164	that may have more than one instance per job include: documentFormatIndex(37) , and
3165	documentFormat(38).
3166	
3167	Examples of document attributes (one instance per document) include: fileName(34), and
3168	documentName(35).
3169	documenti (ame(55).
3170	Examples of required and consumed resource attributes include: pagesRequested(130),
	mediumRequested(170), pagesCompleted(131), and mediumConsumed(171), respectively."
3171	
3172	::= { jmAttributeEntry 1 }
3173	
3174	jmAttributeInstanceIndex OBJECT-TYPE
3175	SYNTAX Integer32(132767)
3176	MAX-ACCESS not-accessible
3177	STATUS current
3178	DESCRIPTION
3179	"A running 16-bit index of the attributes of the same type for each job. For those attributes with
3180	only a single instance per job, this index value SHALL be 1. For those attributes that are a
3181	single value per document, the index value SHALL be the document number, starting with 1 for
3182	the first document in the job. Jobs with only a single document SHALL use the index value of
3183	1. For those attributes that can have multiple values per job or per document, such as
3184	documentFormatIndex(37) or documentFormat(38), the index SHALL be a running index
3185	
	for the job as a whole, starting at 1."
3186	::= { jmAttributeEntry 2 }
3187	
3188	jmAttributeValueAsInteger OBJECT-TYPE
3189	SYNTAX Integer32(-22147483647)
3190	MAX-ACCESS read-only
3191	STATUS current
3192	DESCRIPTION
3193	"The integer value of the attribute. The value of the attribute SHALL be represented as an
3194	integer if the enum description in the JmAttributeTypeTC textual-convention definition has the
3195	tag: 'INTEGER:'.
3196	
3197	Depending on the enum definition, this object value MAY be an integer, a counter, an index, or
3198	an enum, depending on the jmAttributeTypeIndex value. The units of this value are specified
3198	
	in the enum description.
3200	

3201 3202 3203 3204 3205	For those attributes that are accumulating job consumption as the job is processed as specified in the JmAttributeTypeTC textual-convention, SHALL contain the final value after the job completes processing, i.e., this value SHALL indicate the total usage of this resource made by the job.
3206 3207 3208	A monitoring application is able to copy this value to a suitable longer term storage for later processing as part of an accounting system.
3209 3210 3211 3212 3213	Since the agent MAY add attributes representing resources to this table while the job is waiting to be processed or being processed, which can be a long time before any of the resources are actually used, the agent SHALL set the value of the jmAttributeValueAsInteger object to 0 for resources that the job has not yet consumed.
3214 3215 3216 3217	Attributes for which the concept of an integer value is meaningless, such as fileName(34) , jobName , and processingMessage , do <i>not</i> have the 'INTEGER:' tag in the JmAttributeTypeTC definition and so an agent SHALL always return a value of '-1' to indicate 'other' for the value of the jmAttributeValueAsInteger object for these attributes.
3218 3219 3220 3221 3222 3223 3223 3224	For attributes which do have the 'INTEGER:' tag in the JmAttributeTypeTC definition, if the integer value is not (yet) known, the agent either (1) SHALL not materialize the row in the jmAttributeTable until the value is known or (2) SHALL return a '-2' to represent an 'unknown' counting integer value, a '0' to represent an 'unknown' index value, and a '2' to represent an 'unknown(2)' enum value." ::= { jmAttributeEntry 3 }
3225 3226 3227 3228 3229 3230	jmAttributeValueAsOctets OBJECT-TYPE SYNTAX OCTET STRING(SIZE(063)) MAX-ACCESS read-only STATUS current DESCRIPTION
3231 3232 3233	"The octet string value of the attribute. The value of the attribute SHALL be represented as an OCTET STRING if the enum description in the JmAttributeTypeTC textual-convention definition has the tag: 'OCTETS:'.
3234 3235 3236 3237	Depending on the enum definition, this object value MAY be a coded character set string (text), such as ' JmUTF8StringTC' , or a binary octet string, such as ' DateAndTime' .
3237 3238 3239 3240 3241 3242	Attributes for which the concept of an octet string value is meaningless, such as pagesCompleted , do <i>not</i> have the tag 'OCTETS:' in the JmAttributeTypeTC definition and so the agent SHALL always return a zero length string for the value of the jmAttributeValueAsOctets object.
3242 3243 3244 3245 3246 3247	For attributes which do have the 'OCTETS:' tag in the JmAttributeTypeTC definition, if the OCTET STRING value is not (yet) known, the agent either SHALL not materialize the row in the jmAttributeTable until the value is known or SHALL return a zero-length string." ::= { jmAttributeEntry 4 }

3248	Notifications and Trapping			
3249				
3250				
	is hear MIDN stiff satisfy $OD = CT = DENT = U = (is hear MID 2)$			
3251	jobmonMIBNotifications OBJECT IDENTIFIER ::= { jobmonMIB 2}			
3252				
3253				
3254				
3255	Conformance Information			
3256				
3257	jmMIBConformance OBJECT IDENTIFIER ::= { jobmonMIB 3 }			
3258				
3259	compliance statements			
3260	jmMIBCompliance MODULE-COMPLIANCE			
3261	STATUS current			
3262	DESCRIPTION			
3263	"The compliance statement for agents that implement the			
3264	job monitoring MIB."			
3265	MODULE this module			
3266	MANDATORY-GROUPS {			
3267	jmGeneralGroup, jmJobIDGroup, jmJobGroup, jmAttributeGroup }			
3268	Jundenerardroup, Jundoord Group, Jundoo Group, Junitarioace Group			
3269	OBJECT imConoralJabSatNama			
3270	OBJECT jmGeneralJobSetName			
	SYNTAX JmUTF8StringTC (SIZE(08))			
3271	DESCRIPTION			
3272	"Only 8 octets maximum string length NEED be supported by the agent."			
3273				
3274	OBJECT jmJobOwner			
3275	SYNTAX JmJobStringTC (SIZE(016))			
3276	DESCRIPTION			
3277	"Only 16 octets maximum string length NEED be supported by the agent."			
3278	••••••••••••••••••••••••••••••••••••••			
3279	There are no CONDITIONALLY MANDATORY or OPTIONAL groups.			
3280	There are no conditionally with dation i of flowing groups.			
	u_{-} (im)MDC onformance 1)			
3281	::= { jmMIBConformance 1 }			
3282				
3283	jmMIBGroups OBJECT IDENTIFIER ::= { jmMIBConformance 2 }			
3284				
3285	jmGeneralGroup OBJECT-GROUP			
3286	OBJECTS {			
3287	jmGeneralNumberOfActiveJobs, jmGeneralOldestActiveJobIndex,			
3288	jmGeneralNewestActiveJobIndex, jmGeneralJobPersistence,			
3289	jmGeneralAttributePersistence, jmGeneralJobSetName}			
3290	STATUS current			
3291	DESCRIPTION			
3291	"The general group."			
3293	::= { jmMIBGroups 1 }			
3294				
3295	jmJobIDGroup OBJECT-GROUP			
3296	OBJECTS {			

```
jmJobIDJobSetIndex, jmJobIDJobIndex }
3297
            STATUS current
3298
3299
            DESCRIPTION
3300
                  "The job ID group."
3301
            ::= { jmMIBGroups 2 }
3302
3303
       jmJobGroup OBJECT-GROUP
            OBJÉCTS {
3304
                 jmJobState, jmJobStateReasons1, jmNumberOfInterveningJobs,
3305
3306
                 jmJobKOctetsRequested, jmJobKOctetsProcessed, jmJobImpressionsRequested,
3307
                 jmJobImpressionsCompleted, jmJobOwner }
3308
            STATUS current
            DESCRIPTION
3309
                 "The job group."
3310
3311
            ::= { jmMIBGroups 3 }
3312
3313
       jmAttributeGroup OBJECT-GROUP
            OBJECTS {
3314
3315
                 jmAttributeValueAsInteger, jmAttributeValueAsOctets }
            STATUS current
3316
3317
            DESCRIPTION
3318
                 "The attribute group."
3319
            ::= \{ jmMIBGroups \tilde{4} \}
3320
3321
3322
       END
```

3323 **5.** Appendix A - Implementing the Job Life Cycle

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

- 3327 Not all implementations of job submission protocols have all of the states of the job model
- 3328 specified here. The job model specified here is intended to be a superset of most implementations.
- 3329 It is the purpose of the agent to map the particular implementation's job life cycle onto the one
- 3330 specified here. The agent MAY omit any states not implemented. Only the **processing** and
- 3331 **completed** states are required to be implemented by an agent. However, a conforming
- management application SHALL be prepared to accept any of the states in the job life cycle
- 3333 specified here, so that the management application can interoperate with any conforming agent.
- 3334 The job states are intended to be user visible. The agent SHALL make these states visible in the
- 3335 MIB, but only for the subset of job states that the implementation has. Some implementations
- 3336 MAY need to have sub-states of these user-visible states. The jmJobStateReasons1 object and
- the jobStateReasonsN (N=2..4) attributes can be used to represent the sub-states of the jobs.
- Job states are intended to last a user-visible length of time in most implementations. However, some jobs may pass through some states in zero time in some situations and/or in some
- 3340 implementations.
- 3341 The job model does not specify how accounting and auditing is implemented, except to assume
- that accounting and auditing logs are separate from the job life cycle and last longer than job
- antries in the MIB. Jobs in the completed, aborted, or canceled states are not logs, since jobs in
- these states are accessible via SNMP protocol operations and SHALL be removed from the Job
- 3345 Monitoring MIB tables after a site-settable or implementation-defined period of time. An
- accounting application MAY copy accounting information incrementally to an accounting log as a
- job processes, or MAY be copied while the job is in the **canceled**, **aborted**, or **completed** states,
- depending on implementation. The same is true for auditing logs.

3349The jmJobState object specifies the standard job states. The normal job state transitions3350are shown in the state transition diagram presented in Table 1.

3351 6. APPENDIX B - Support of the Job Submission ID in Job Submission 3352 Protocols

- 3353 This appendix lists the job submission protocols that support the concept of a job
- submission ID and indicates the attribute used in that job submission protocol.

3355 6.1 Hewlett-Packard's Printer Job Language (PJL)

Hewlett-Packard's Printer Job Language provides job-level printer control and printer
status information to applications. The PJL JOB command is used at the beginning of a
print job and can include options applying only to that job. A PJL JOB command option
has been defined to facilitate passing the JobSubmissionID with the print job, as required
by the Job Monitoring MIB. The option is of the form:

3361 3362

3363

SUBMISSIONID = "id string"

Where the "id string" is a string and SHALL be enclosed in double quotes. The format is as described for the **jmJobSubmissionID** object.

3366 The entire PJL JOB command with the optional parameter would be of the form:

3367 3368 3369

@PJL JOB SUBMISSIONID = "id string"

See "Printer Job Language Technical Reference Manual", part number 5021-0328, from
Hewlett-Packard for complete information on the PJL JOB command and the Printer Job
Language.

3373 NOTE - Some PJL implementations wrap a banner page as a PJL job around a job

3374 submitted by a client. In this case, there will be two job submission ids. The outer one

being the one with the banner page and the inner one being the original user's job. The

3376 agent SHALL use the last received job submission ID for the jmJobSubmissionID index,

so that the original user's job submission ID will be used, not the banner page job ID.

3378 6.2 ISO DPA

3379 The ISO 10175 Document Printing Application (DPA) protocol specifies the "job-client-

id" attribute that allows the client to supply a text string ID for each job.

3381 7. References

- 3382 [char-set policy] Harald Avelstrand, "IETF Policy on Character Sets and Language", June
 3383 1997. Latest draft: <draft-avelstrand-charset-policy-00.txt>
- [GB2312] GB 2312-1980, "Chinese People's Republic of China (PRC) mixed one byte
 and two byte coded character set"
- 3386 [hr-mib] P. Grillo, S. Waldbusser, "Host Resources MIB", RFC 1514, September 1993

[iana] J. Reynolds, and J. Postel, "Assigned Numbers", STD 2, RFC 1700, ISI, October1994.

3389 [IANA-charsets] Coded Character Sets registered by IANA and assigned an enum value 3390 for use in the **CodedCharSet** textual convention imported from the Printer MIB. See ftp://ftp.isi.edu/in-notes/iana/assignments/character-sets 3391 3392 [iana-media-types] IANA Registration of MIME media types (MIME content 3393 types/subtypes). See ftp://ftp.isi.edu/in-notes/iana/assignments/ 3394 [ISO 646] ISO/IEC 646:1991, "Information technology -- ISO 7-bit coded character set 3395 for information interchange", JTC1/SC2. 3396 [ISO 8859] ISO/IEC 8859-1:1987, "Information technology -- 8-bit single byte coded 3397 graphic character sets - Part 1: Latin alphabet No. 1, JTC1/SC2." 3398 [ISO 2022] ISO/IEC 2022:1994 - "Information technology -- Character code structure 3399 and extension techniques", JTC1/SC2. 3400 [ISO-10646] ISO/IEC 10646-1:1993, "Information technology -- Universal Multiple-3401 Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane, 3402 JTC1/SC2. 3403 [iso-dpa] ISO/IEC 10175 Document Printing Application (DPA). See 3404 ftp://ftp.pwg.org/pub/pwg/dpa/ [ipp-model] Internet Printing Protocol (IPP), work in progress on the IETF standards 3405 3406 track. See draft-ietf-ipp-model-01.txt. See also http://www.pwg.org/ipp/index.html [JIS X0208] JIS X0208-1990, "Japanese two byte coded character set." 3407 3408 [mib-II] MIB-II, RFC 1213. 3409 [print-mib] The Printer MIB - RFC 1759, proposed IETF standard. Also an Internet-3410 Draft on the standards track as a draft standard: draft-ietf-printmib-mib-info-02.txt 3411 [req-words] S. Bradner, "Keywords for use in RFCs to Indicate Requirement Levels", 3412 RFC 2119, March 1997. 3413 [rfc 2130] C. Weider, C. Preston, K. Simonsen, H. Alvestrand, R. Atkinson, M. Crispin, 3414 and P. Svanberg, "The Report of the IAB Character Set Workshop held 29 Feb-1 March, 3415 1997", April 1997, RFC 2130. 3416 [SMIv2-TC] J. Case, et al. "Textual Conventions for Version 2 of the Simple Network 3417 Management Protocol (SNMPv2)", RFC 1903, January 1996. 3418 [tipsi] IEEE 1284.1, Transport-independent Printer System Interface (TIPSI). 3419 [URI-spec] Berners-Lee, T., Masinter, L., McCahill, M., "Uniform Resource Locators 3420 (URL)", RFC 1738, December, 1994. 3421 [US-ASCII] Coded Character Set - 7-bit American Standard Code for Information 3422 Interchange, ANSI X3.4-1986.

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

3425	8. Author's Addresses
3426	Ron Bergman
3427	Dataproducts Corp.
3428	1757 Tapo Canyon Road
3429	Simi Valley, CA 93063-3394
3430	
3431	Phone: 805-578-4421
3432	Fax: 805-578-4001
3433	Email: rbergman@dpc.com
3434	
3435	
3436	Tom Hastings
3437	Xerox Corporation, ESAE-231
3438	701 S. Aviation Blvd.
3439	El Segundo, CA 90245
3440	
3441	Phone: 310-333-6413
3442	Fax: 310-333-5514
3443	EMail: hastings@cp10.es.xerox.com
3444	
3445	
3446	Scott A. Isaacson
3447	Novell, Inc.
3448	122 E 1700 S
3449	Provo, UT 84606
3450	
3451	Phone: 801-861-7366
3452	Fax: 801-861-4025
3453	EMail: scott_isaacson@novell.com
3454	
3455	
3456	Harry Lewis
3457	IBM Corporation
3458	6300 Diagonal Hwy
3459	Boulder, CO 80301
3460	
3461	Phone: (303) 924-5337

3462	Fax:
3463	Email: harryl@us.ibm.com
3464	·
3465	
3466	Send comments to the printmib WG using the Job Monitoring Project (JMP)
3467	Mailing List: jmp@pwg.org
3468	
3469	To learn how to subscribe, send email to: jmp-request@pwg.org
3470	
3471	For further information, access the PWG web page under "JMP":
3472	http://www.pwg.org/
3473	
3474	Other Participants:
3475	Chuck Adams - Tektronix
3476	Jeff Barnett - IBM
3477	Keith Carter, IBM Corporation
3478	Jeff Copeland - QMS
3479	Andy Davidson - Tektronix
3480	Roger deBry - IBM
3481	Mabry Dozier - QMS
3482	Lee Ferrel - Canon
3483	Steve Gebert - IBM
3484	Robert Herriot - Sun Microsystems Inc.
3485	Shige Kanemitsu - Kyocera
3486	David Kellerman - Northlake Software
3487	Rick Landau - Digital
3488	Harry Lewis - IBM
3489	Pete Loya - HP
3490	Ray Lutz - Cognisys
3491	Jay Martin - Underscore
3492	Mike MacKay, Novell, Inc.
3493	Stan McConnell - Xerox
3494	Carl-Uno Manros, Xerox, Corp.
3495	Pat Nogay - IBM
3496	Bob Pentecost - HP
3497	Rob Rhoads - Intel
3498	David Roach - Unisys
3499	Hiroyuki Sato - Canon
3500	Bob Setterbo - Adobe
3501	Gail Songer, EFI

- 3502 Mike Timperman Lexmark
- 3503 Randy Turner Sharp
- 3504 William Wagner Digital Products
- Jim Walker Dazel
- 3506 Chris Wellens Interworking Labs
- 3507 Rob Whittle Novell
- 3508 Don Wright Lexmark
- 3509 Lloyd Young Lexmark
- 3510 Atsushi Yuki Kyocera
- 3511 Peter Zehler, Xerox, Corp.

3512 **9. INDEX**

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

		3550	jmGeneralNewestActiveJobIndex	75
3517	—C—	3551	jmGeneralNumberOfActiveJobs	74
5517	0	3552	jmGeneralOldestActiveJobIndex	
3518	colorantConsumed	58 3553	jmJobIDJobIndex	
3519	colorantRequested		jmJobIDJobSetIndex	
	1	3555	jmJobImpressionsCompleted	
2520	D	3556	jmJobImpressionsRequested	
3520	—D—	3557	jmJobIndex	
3521	deviceNameRequested		jmJobKOctetsProcessed	
3522	documentCopiesCompleted	55 3559	jmJobKOctetsRequested	
3523	documentCopiesRequested	54 3560	jmJobOwner	
3524	documentFormat	51 3561	JmJobServiceTypesTC	
3525	documentFormatIndex		JmJobSourcePlatformTypeTC	
3526	documentName	50 3563	jmJobState	
		3564	jmJobStateReasons1	
2507	— F —	3565	JmJobStateReasons1TC	
3527	— r —	3566	JmJobStateReasons2TC	
3528	fileName	50 3567	JmJobStateReasons3TC	
3529	finishing		JmJobStateReasons4TC	
3530	fullColorImpressionsCompleted	56 3569	JmJobStateTC	
	r in r	3570	JmJobStringTC	
2521	—H—	3571	jmJobSubmissionID	
3531	— n —	3572	JmJobSubmissionTypeTC	
3532	highlightColorImpressionsCompleted	56 3573	JmMediumTypeTC	
	0 0 · · · · · · · · · · ·	3574	jmNumberOfInterveningJobs	
3533	— I —	3575	JmPrinterResolutionTC	37
5555	— I —	3576	JmPrintQualityTC	37
3534	impressionsCompletedCurrentCopy	56 3577	JmTimeStampTC	34
3535	impressionsInterpreted	56 35 78	JmTonerEconomyTC	
3536	impressionsSentToDevice	56 3579	JmUTF8StringTC	
3537	impressionsSpooled	56 3580	jobAccountName	
		3581	jobCodedCharSet	46
3538	J	3582	jobComment	51
	-	3583	jobCompletedTime	
3539	jmAttributeInstanceIndex	85 3584	jobCopiesCompleted	
3540	jmAttributeTypeIndex	85 3585	jobCopiesRequested	
3541	JmAttributeTypeTC	45 3586	jobHold	
3542	jmAttributeValueAsInteger	86 3387	jobHoldUntil	
3543	jmAttributeValueAsOctets	87 3588	jobKOctetsTransferred	
3544	JmBooleanTC	38 3389	jobName	
3545	JmFinishingTC	35 3590	jobOriginatingHost	
3546	jmGeneralAttributePersistence	75 3591	jobPriority	
3547	jmGeneralJobPersistence	75 3592	jobProcessAfterDateAndTime	
3548	jmGeneralJobSetIndex	73 3593	jobProcessingCPUTime	
3549	jmGeneralJobSetName	76 3594	jobServiceTypes	48

Job Monitoring MIB, V0.85 Aug 8, 1997

3595	jobSourceChannelIndex	р
3596	jobSourcePlatformType	p
3597	jobStartedBeingHeldTime	p
3598	jobStartedProcessingTime	p
3599	jobStateReasons2	p
3600	jobStateReasons3	p
3601	jobStateReasons4	p
3602	jobSubmissionTime 59	
3603	jobSubmissionToServerTime	
3604	—M — 3623	q
3605	mediumConsumed	
3606	mediumRequested	
2000	3625	S
3607	— N — 3626	sl
3007	362/	sl
3608	numberOfDocuments	sl
	3629	si
3609	-0- 3630	sı
	3631	sı
3610	other	
3611	outputBin	
3612	P	to
	- 3634	to
3613	pagesCompleted	to
3614	pagesCompletedCurrentCopy 57 3636	to
3637		

pagesRequested	57
physicalDevice	
printerResolutionRequested	
printerResolutionUsed	54
printQualityRequested	53
printQualityUsed	53
processingMessage	46

Q

serverAssignedJobName	47
sheetsCompleted	
sheetsCompletedCurrentCopy	57
sheetsRequested	57
sides	53
submittingApplicationName	49
submittingServerName	49

—T—

tonerDensityRequested	54
tonerDensityUsed	
tonerEcomonyRequested	
tonerEcomonyUsed	

3637