	Job Monitoring MIB, V0. <u>871</u> April 4, 1997	
1	Job Monitoring MIB	
2 3 4 5 6 7 8 9 10 11 12 13 14 15	From: Tom Hastings Date: 04/0403/20/97 Version: 0. <u>8</u> 71 File: ftp://ftp.pwg.org/pub/jmp/mibs/jmp-mib.doc .pdfjmp-mibr.doc .pdf .pdr Status: FourthThird draft MIB that corresponds to the changes agreed to at the JMP meeting, 04/04/97 in Austin. Harry Lewis's changes to eliminate the Queue and Completed tables and to replace the Job table with the Job ID and Job State table have been incorporated.sixth draft spec as agreed at the 02/07/97 JMP meeting and subsequent telecons. This is version 0.71. There are just a few changes from version 0.7, mostly editorial. See the change history. The Internet-Draft was not posted in time and with these changes, we will not present any MIB document at the IETF meeting on 04/08/97 in Memphis. Instead we will present slides on the current status explaining the tables, which are: General Job ID. Job State, and Attributes	
16 17 18	The MIB has been greatly simplified so that now there are only <u>1327</u> objects in the MIB÷ <u>21 mandatory and 6 conditionally mandatory</u> . There are 57 attributes, of which only 7 are <u>mandatory</u> .	
19 20 21	I've removed the issues from the document and placed them in a separate document: issues.doc .pdf. There are very few issues remaining. I've added a few issues from the e-mail since the last telecon.	
22 23 24	The actual specifications of each object needs line-by-line review. We did <i>not</i> have time for such review at the $11/08/96$ or the $01/08/97$ meeting as indicated in the minutes. The group wanted to wait until this specification is re-formatted into a MIB.	
25 26 27 28 29 30 31 32 33	The greatly simplified specifications of each object is derived from the ISO DPA attribute specifications in most cases. I've moved the full ISO DPA specifications to <u>a separate</u> <u>document</u> <u>an Appendix</u> . Revision marks show the agreements reached at the November meeting where we were able to finish the entire document. I've indicated ISSUES in <u>a</u> <u>separate document</u> <u>the text</u> that we have identified as issues but have not resolved. These issues are also listed at the end of the Table of Contents with the page number of the issue. I've also copied in-map-summ.doc into <u>another this</u> document and moved it to an appendix so we can <u>more easily</u> compare the Job Monitoring objects with the job submission protocols and keep the object names updated in that summary.	
34 35 36	We moved more objects into the Resource Table, now called the Attribute Table, since more than resources are in it. I've not used revision marks for such moves, but only for changes within each description of what had been an object and what now is an enum.	
37	I've moved Ron's re-written introduction into the document.	

38

39	INTERNET-DRAFT
40	Ron Bergman
41	Dataproducts Corp.
42	Tom Hastings
43	Xerox Corporation
44	Scott Isaacson
45 46	Novell, Inc.
40 17	IBM Com
48	April March 1997
49	
50	
51	Job Monitoring MIB - V0.7
52	<draft-ietf-printmib-job-monitor-00.txt></draft-ietf-printmib-job-monitor-00.txt>
53	Expires Oct 4, 1997
54 55	
56	Status of this Memo
57 58 59	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.
60 61 62 63	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."
64 65 66	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),
67	ds.internic.net (US East Coast), or ftp.isi.edu (US West Coast).
68	Abstract
69	This Internet-Draft specifies a set of <u>13</u> SNMP MIB objects for (1) monitoring the
70	status and progress of print jobs (2) obtaining resource requirements before a job
71	is processed, (3) monitoring resource consumption while a job is being processed
72	and (4) collecting resource accounting data after the completion of a job. This
73	MIB is intended to be implemented (1) in <u>a printers</u> or (2) in a server that supports
74	one or more printers. Use of the object set is not limited to printing. However,
75	support for services other than printing is outside the scope of this Job Monitoring
/6 77	MIB. Future extensions to this MIB may include, but are not limited to, fax
11	machines and scanners.

78		
79	TABLE OF CONTENTS	
80	1. INTRODUCTION	8
81	1.1 Types of Information in the MIB	8
82	1.2 Types of Job Monitoring Applications	9
83	2. TERMINOLOGY AND JOB MODEL	10
84	3. SYSTEM CONFIGURATIONS FOR THE JOB MONITORING MIB	20
85	3.1 Configuration 1 - client-printer	20
86	3.2 Configuration 2 - client-server-printer - agent in the server	22
87	3.2 Configuration 3 - client-server-printer - client monitors printer agent and server	24
88	4. CONFORMANCE CONSIDERATIONS	26
89	3.2 Conformance Terminology	26
90	3.3 Agent Conformance Requirements	26
91	3.4 Job Monitoring Application Conformance Requirements	27
92	4. JOB IDENTIFICATION	27
93	5. INTERNATIONALIZATION CONSIDERATIONS	28
94	6. IANA CONSIDERATIONS	28
95	6.1 IANA Registration of enums	28
96	6.2 IANA Registration of bit string values	29
97	7. SECURITY CONSIDERATIONS	30
98	7.1 Read-Write objects	30
99	7.2 Read-Only Objects In Other User's Jobs	30
100	8. RETURNING OBJECTS WITH NO VALUE IN MANDATORY GROUPS	30

Job Monitoring MIB, V0.<u>871</u> April 4, 1997

101	9. NOTIFICATION AND TRAPS	30
102	10. MIB SPECIFICATION	30
103	Textual conventions for this MIB module	32
104	JmTimeTC - simple time in seconds	32
105	JmTimeIntervalTC - simple time interval in seconds	32
106	JmJobStateTC - job state definitions	32
107	JmAttributeTypeTC - attribute type definitions	37
108	other	39
109	Job State attributes	39
110	jobState (mandatory)	39
111	jobStateAssociatedValue	39
112	jobStateReasons	40
113	numberOfInterveningJobs (mandatory)	41
114	deviceAlertCode (mandatory)	41
115	processingMessage	41
116	Job Identification attributes	41
117	jobName	41
118	jobServiceTypes	42
119	jobOwner	43
120	jobAccountName	43
121	jmJobDeviceNameOrQueueRequested	43
122	jobSourceChannelIndex	43
123	physicalDeviceIndex	44
124	physicalDeviceName	44
125	fileName	44
126	documentName	44
127	jobComment	44
128	Job Parameter attributes	45
129	jobPriority	45
130	jobProcessAfterDateAndTime	45
131	outputBinIndex	46
132	outputBinName (mandatory)	46
133	sides	46
134	documentFormatIndex	46
135	documentFormatEnum	47
136	Resource attributes (requested and consumed)	47
137	jobCopiesRequested	47
138	jobCopiesCompleted	47
139	documentCopiesRequested	47
140	jobKOctetsRequested (mandatory)	48
141	jobKOctetsCompleted (mandatory)	49
142	Impression attributes	49
143	impressionsSpooled	50
144	impressionsSentToDevice	50
145	impressionsInterpreted	50
146	impressionsRequested (mandatory)	50
147	impressionsCompleted (mandatory)	50
148	impressionsCompletedCurrentCopy	50
149	Page attributes	50
150	pagesRequested	51

Job Monitoring MIB, V0.871 April 4, 1997

[Page 6]

151	pagesCompleted	51
152	pagesCompletedCurrentCopy	51
153	Sheet attributes	51
154	sheetsRequested	51
155	sheetsCompleted	51
156	sheetsCompletedCurrentCopy	51
157	mediumRequested	51
158	mediumConsumed	51
159	colorantRequestedIndex	52
160	colorantRequestedName	52
161	colorantConsumedIndex	52
162	colorantConsumedName	52
163	Time attributes	52
164	iobSubmissionDateAndTime	53
165	jobSubmissionTime	53
166	iobStartedBeingHeldTime	53
167	jobStartedProcessingDateAndTime	53
168	iohStartedProcessingTime	53
169	jobCompletedDateAndTime	53
170	jobCompletedTime	54
171	processingCPI Time	54
172	ImJohServiceTypesTC - bit encoded job service type definitions	54
173	ImJobStateReasonsTC - additional information about job states	56
175	shisobstate (casons i C - additional information about job states	50
174	The General Group (Mandatory)	68
175	imGeneral lobSetName	69
176	imGeneralIobPersistence	69
177	imGeneralAttributePersistence	69
178	imGeneralNumberOfActive Jobs	70
179	imGeneralOldestActiveJobIndex	70
180	imGeneralNewestActiveIobIndex	71
100	Jinoonoran (e wood ted yeb oonder	, 1
181	The Job ID Group (Mandatory)	77
182	jmJobSubmissionIDIndex	78
183	jmJobSetIndex	79
184	jmJobIndex	79
	5	
185	The Job State Group (Mandatory)	79
186	jmJobState	84
187	jmJobStateKOctetsCompleted	86
188	jmJobStateImpressionsCompleted	86
189	jmJobStateAssociatedValue	86
	•	
190	The Attribute Group (Mandatory)	87
191	jmAttributeTypeIndex	89
192	jmAttributeInstanceIndex	89
193	jmAttributeValueAsInteger	90
194	jmAttributeValueAsOctets	90
195	11. JOB LIFE CYCLE	95
196	12. BIBLIOGRAPHY	97

197	13. AUTHOR'S ADDRESSES	97
198	14. CHANGE HISTORY (NOT TO BE INCLUDED IN THE INTERNET DR	AFT)100
199	14.1 Changes to version 0.7, dated 3/13/97 to make version 0.71, dated 3/26/97	100
200	14.2 Changes to version 0.6, dated 1/23/97 to make version 0.7, dated 3/13/97	100
201	15. INDEX	104

203 **Job Monitoring MIB**

204 1. Introduction

The Job Monitoring MIB consists of a 5-object General Group, a 2-object Job Submission 205 ID Group, a 4-object Job State Group, and a 2-object Attribute Group. Each group is a 206 207 table. The General Group contains general information that applies to all jobs in a job set. 208 The Job Submission ID table maps the job submission ID that the client uses to identify a 209 job to the jmJobIndex that the Job Monitoring Agent uses to identify jobs in the Job State 210 and Attribute tables. The Job State table contains the job state and copies of three salient 211 attributes for each job's current state. The Attribute table consists of multiple entries per 212 job that specify (1) job and document identification and parameters, (2) requested 213 resources, and (3) consumed resources during and after job processing/printing. The Job 214 Monitoring MIB contains a set of objects for (1) monitoring the status and progress of 215 print jobs, (2) obtaining resource requirements before a job is processed, (3) monitoring 216 resource consumption while a job is being processed and (4) collecting resource 217 accounting data after the completion of a job. This MIB is intended to be implemented in 218 printers or a server that supports one or more printers. Use of the object set is not limited 219 to printing. However, support for services other than printing is outside the scope of this 220 Job Monitoring MIB. Future extensions to this MIB may include, but are not limited to, 221 fax machines and scanners. 222 The Job Monitoring MIB is intended to be instrumented by an agent within a printer or the 223 first server closest to the printer, where the printer is either directly connected to the 224 server only or the printer does not contain the job monitoring MIB agent. It is 225 recommended that implementations place the SNMP agent as close as possible to the 226 processing of the print job. This MIB applies to printers with and without spooling 227 capabilities. This MIB is designed to be compatible with most current commonly-used job

228 submission protocols. In most environments that support high function job submission/job

229 control protocols, like ISO DPA, those protocols would be used to monitor and manage 230

print jobs rather than using the Job Monitoring MIB.

231 **1.1** Types of Information in the MIB

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

- 233 Models in the Printer MIB (Refer to RFC 1759, Appendix D - Roles of Users).
- 234 User:
- 235 Provide the ability to identify the least busy printer. The user will be able to 236 determine the number and size of jobs waiting for each printer. No attempt is 237
- made to actually predict the length of time that jobs will take.
- 238 Provide the ability to identify the current status of the job (user queries).
- 239 Provide a timely notification that the job has completed and where it can be 240 found.

Provide error and diagnostic information for jobs that did not successfully complete.
Operator:
Provide a presentation of the state of all the jobs in the print system.
Provide the ability to identify the user that submitted the print job.
Provide the ability to identify the resources required by each job.
Provide the ability to define which physical printers are candidates for the print job.
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.
Capacity Planner:
Provide the ability to determine printer utilization as a function of time.
Provide the ability to determine how long jobs wait before starting to print.
Accountant:
Provide information to allow the creation of a record of resources consumed and printer usage data for charging users or groups for resources consumed.
Provide information to allow the prediction of consumable usage and resource need.
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 networked devices without unnecessary overhead or complexity, while also providing for higher end systems and devices.
1.2 <u>Types of Job Monitoring Applications</u>
The Job Monitoring MIB is designed for the following types of monitoring applications:
 monitor a single job starting when the job is submitted and finishing a defined period after the job completes. The Job Submission ID table provides the map to find the specific job to be monitored.
 monitor all active of the jobs in a queue, which is generalized 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 co-located with the printer to provide an enhanced console capability.

278 279 280 281	3. <u>collect resource usage for accounting or system utilization purposes that</u> copy the completed job statistics to an accounting system. It is recognized that depending on accounting programs to copy MIB data during the job-retention period is somewhat unreliable, since the accounting program may not be running (or may
282	have crashed). Such a program is expected to keep a shadow copy of the entire
283	Job Attribute table including cancelled and completed jobs which the program
284	updates on each poining cycle. Such a program poils at the rate of the persistence
285	of the Attribute table. The design is not optimized to help such an application determine which is a completed or conceled. Instead, the application shall
280	determine which jobs are completed of canceled. Instead, the application shall query each job that the application's shadow copy shows was not complete or
207	canceled at the previous poll cycle to see if it is now complete or canceled plus
288	any new jobs that have been submitted.
290 291	The MIB provides job resource accounting information after the printer has finished printing the job. This resource accounting information is intended to be used by:
292 293	 A management station that is co-located with the printer to provide an enhanced console capability.
294 295 296	 End user job monitoring programs that provide status on progress and completion of jobs during the complete life cycle of the job, including a defined period after the job completes.
297 298 299 300	• System accounting programs that copy the completed job statistics to an accounting system. It is recognized that depending on accounting programs to copy MIB data during the job-retention period is somewhat unreliable, since the accounting program may not be running (or may have crashed).
301 302 303 304 305 306	The MIB provides a set of objects that represent a compatible subset of job and document attributes of the ISO DPA standard, so that coherence is maintained between the two protocols and information presented to end users and system operators. However, the job monitoring MIB is intended to be used with printers that implement other job submitting and management protocols, such as IEEE 1284.1 (TIPSI), as well as with ones that do implement ISO DPA. So nothing in the job monitoring MIB shall require implementation

- 307 of the ISO DPA protocol.
- 308 The MIB is designed so that an additional MIB(s) can be specified in the future for 309 monitoring multi-function (scan, FAX, copy) jobs as an augmentation to this MIB.

310 2. Terminology and Job Model

- This section defines the terms that are used in this specification and the general model for jobs.
- 313 NOTE Existing systems use conflicting terms, so these terms are drawn from the ISO
- 314 10175 Document Printing Application (DPA) standard. For example, PostScript
- 315 systems use the term *session* for what we call a *job* in this specification and the term
- 316 *job* to mean what we call a *document* in this paper. PJL systems use the term ..

Job Monitoring MIB, V0.<mark>871</mark>

317 A *job* is a unit of work whose results are expected together without interjection of

318 unrelated results. A *client* is able to specify *job instructions* that apply to the job as a

319 whole. Proscriptive instructions specify how, when, and where the job is to be printed.

320 Descriptive instructions describe the job. A job contains one or more *documents*.

A *job set* is a set of jobs that are queued and scheduled together according to a specified scheduling algorithm for a specified device or set of devices. For implementations that

323 embed the SNMP agent in the device, the MIB job set normally represents *all* the jobs

324 known to the device, so that the implementation only implements a single job set which

- 325 <u>may be identified with a hard-coded value 1</u>. If the SNMP agent is implemented in a
- server that controls one or more devices, each MIB job set represents a job queue for (1)
- a specific device or (2) set of devices, if the server uses a single queue to load balance
 between several devices. Each job set is disjoint; no job shall be represented in more than
 one MIB job set.

A *document* is a sub-section within a job. A document contains print data and *document*

instructions that apply to just the document. The *client* is able to specify document

instructions separately for each document in a job. Proscriptive instructions specify how

the document is to be processed and printed by the *server*. Descriptive instructions

- describe the document. Server implementation of more than one document per job isoptional.
- A *client* is the network entity that *end users* use to submit jobs to *spoolers*, *servers*, or
 printers and other *devices*, depending on the configuration, using any job submission
 protocol.

A *server* is a network entity that accepts jobs from clients and in turn submits the jobs to *printers* and other *devices*. A server may be a printer *supervisor* control program, or a print *spooler*.

342 A *device* is a hardware entity that (1) interfaces to humans in human perceptible means,

343 such as produces marks on paper, scans marks on paper to produce an electronic

- representations, or writes CD-ROMs or (2) interfaces to a network, such as sends FAX data to another FAX device
- 345 data to another FAX device.
- 346 A *printer* is a *device* that puts marks on media.
- A *supervisor* is a server that contains a control program that controls a printer or other
 device. A supervisor is a client to the printer or other device.
- 349 A *spooler* is a server that accepts jobs, spools the data, and decides when and on which
- 350 printer to print the job. A spooler is a client to a printer or a printer supervisor, depending 351 on implementation.
- 352 *Spooling* is 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.
- 354 *Queuing* is the act of a *device* or *server* of ordering (queuing) the jobs for the purposes of 355 scheduling the jobs to be processed.

Job Monitoring MIB, V0.<mark>871</mark>

- 356 A *monitor* or *job monitoring application* is the network entity that End Users, System
- 357 Operators, Accountants, Asset Managers, and Capacity Planners use to monitor jobs using
- 358 SNMP. A monitor may be either a separate application or may be part of the client that 359 also submits jobs.
- An *agent* is the network entity that accepts SNMP requests from a *monitor* andimplements the Job Monitoring MIB.
- 362 A *proxy* is an agent that acts as a concentrator for one or more other agents by accepting
- 363 SNMP operations on the behalf of one or more other agents, forwarding them on to those
- 364 other agents, gathering responses from those other agents and returning them to the
- 365 original requesting monitor.
- 366 A *user* is a person that uses a client or a monitor.
- 367 An *end user* is a user that uses a client to submit a print job.
- 368 A *system operator* is a user that uses a monitor to monitor the system and carries out tasks 369 to keep the system running.
- 370 A system administrator is a user that specifies policy for the system.
- 371 A *job instruction* is an instruction specifying how, when, or where the job is to be
- 372 processed. Job instructions may be passed in the job submission protocol or may be
- 373 embedded in the document data or a combination depending on the job submission
- 374 protocol and implementation.
- A *document instruction* is an instruction specifying how to process the document.
- 376 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,
- depending on the job submission protocol and implementation.
- 379 An SNMP information object is a name, value-pair that specifies an action, a status, or a
- condition in an SNMP MIB. Objects are identified in SNMP by an OBJECT
 IDENTIFIER
- 381 <u>IDENTIFIER</u>.
- 382 An *attribute* is a name, value-pair that specifies an instruction, a status, or a condition <u>ofin</u>
- a job or a document that has been submitted to a server or device in a job submission
- 384 **protocol**. A <u>particular n</u>-attribute need not be present in each job instance. In other
- 385 words, attributes are present in a job instance only when there is a need to express the
- value, either because (1) the client supplied a value in the job submission protocol, (2) the
- 387 document data contained an embedded attribute, or (3) the server or device supplied a
- 388 <u>default value</u>. The term "attribute" will be used when discussing a *job instruction* or a
- 389 *document instruction* in a job submission protocol that is not embedded in the document
- 390 data. <u>An agent shall represent an attribute as The term "attribute" will also be used for an</u>
- 391 <u>entry (row) in the attribute table in this MIB in which entries are present only when</u>
- 392 necessary. <u>Attributes are identified in this MIB by an enum. The term "information object"</u>
- 393 or "object" for short will be used in discussing the MIB. In other words, the server or
- 394 printer accepts jobs via a job submission protocol that contains job and document
- 395 attributes and the SNMP agent instruments the job by returning the equivalent, possibly
- 396 transformed, job and document attributes as MIB objects in response to SNMP Get

Job Monitoring MIB, V0.<u>871</u>

- 397 requests. The agent may also represent job and document instructions that are embedded
 398 in the document data as MIB objects, depending on implementation.
- *Job monitoring* using SNMP is (1) identifying jobs within the serial streams of data being
- 400 processed by the server, printer or other devices, (2) creating "rows" in the job table for
- 401 each job, and (3) recording information, known by the agent, about the processing of the402 job in that "row".
- 403 *Job accounting* is recording what happens to the job during the processing and printing of404 the job.
- 405 The job model has the following states:

State	Summary Description
1. unknown	The state of the job is not known to the agent or is unknowable, or the job is not yet created or has just been purged.
2. preProcessi ng	The job has been created on the server or device but the submitting client is in the process of adding additional job components and no documents have started processing. The job maybe in the process of being checked by the server/device for attributes, defaults being applied, a device being selected, etc.
3. held	The job is not yet a candidate for processing for any number of reasons. The reasons are represented as bits in the jmJobStateReasons object. Some reasons are used in other states to give added information about the job state. See the JmJobStateReasonsTC textual convention for the specification of each reason and in which states the reasons may be used.
4. pending	The job is a candidate for processing, but is not yet processing.
5. processing	The job is using one or more document transforms which include purely software processes, such as interpreting a PDL, and hardware devices.
6. needsAtten tion	The job is using one or more devices, but has encountered a problem with at least one device that requires human intervention before the job can continue using that device. Examples include running out of paper or a paper jam.
	Usually devices indicate their condition in human readable form locally at the device. The management application can obtain more complete device status remotely by querying the appropriate device MIB using the job's jmDeviceIndex object in the Job Monitoring MIB.
	NOTE - Instead of the needsAttention job state, ISO DPA uses the multi-valued printer-state-of-printers-assigned job attribute, so that the state of each device that a job is using can be accurately represented. However, for the Job Monitoring MIB, the simpler approach is used of adding a single needsAttention job state if any device that the job is using needs attention and relying on the device MIB for more information.
7. paused	The job has been indefinitely suspended by a client issuing an operation to suspend the job so that other jobs may proceed using the same devices. The client may issue an operation to resume the

406 Table: Job Object Life Cycle Summary

State	Summary Description
	paused job at any time, in which case the server or printer places the job in the held or pending states and the job is eventually resumed at the point where the job was paused.
8. interrupted	The job has been interrupted while processing by a client issuing an operation that specifies another job to be run instead of the current job. The server or printer will automatically resume the interrupted job when the interrupting job completes.
9. terminating	The job is in the process of being terminated by the server or printer, either because the client canceled the job or because a serious problem was encountered by a document transform while processing the job. The job's jmJobStateReasons object shall contain the reasons that the job was terminated.
10. retained	The job is being retained by the server or printer after processing and all of the media have been successfully stacked in the output bin(s).
	The job (1) has completed successfully or with warnings or errors, (2) has been aborted while printing by the server/device, or (3) has been cancelled by the submitting user or operator before or during processing. The job's jmJobStateReasons object shall contain the reasons that the job has entered the retained state.
	While in the retained state, all of the job's document data (and submitted resources, if any) are retained by the server or device; thus a client could issue an operation to resubmit the job (or a copy of the job) while the job is in the retained state.
	The retained state is conditionally mandatory. Implementations that do <i>not</i> retain jobs after they are finished processing such that the client could request that the job be repeated (or resubmitted), need not implement the retained state.
11. completed	The job has (1) completed processing, (2) all of the media have been successfully stacked in the output bin(s) and (3) the server/device is keeping the job in summary form for a site settable period for purposes of aiding operators and users to determine the disposition of users' jobs.
	The job (1) has completed successfully or with warnings or errors, (2) has been aborted while printing by the server/device, or (3) has been cancelled by the submitting user or operator before or during processing. The job's jmJobStateReasons object shall contain the reasons that the job has entered the completed state.

State	Summary Description
	While in the completed state, a job's document data (and submitted resources if any) need not be retained by the server; thus a job in the completed state could not be reprinted. The length of time that a job may be in this state, before transitioning to unknown , is implementation dependent. However, servers that implement the completed job state shall retain all of the job's Job Monitoring MIB objects, except the jmQueueGroup objects, so that a management application accounting program can copy them to an accounting log.

407 408	There are two approaches that implementers may use to address the problems of the end- user using the Job Monitoring MIB:		
409 410	 The client also supports SNMP and the Job Monitoring MIB for status/notification to the submitting user 		
411 412 413	 The monitor supports SNMP and the Job Monitoring MIB for status/notification to any user, including the job submitting end user; for example, the Windows Print Manager. 		
414 415 416 417 418 410	The following diagram illustrates the relationships between the defined entities.		
419 420 421 422 423			
423 424 425 426			
427 428 429			

430 Figure- Relationship between client, printer/server, management station, and agent

Job Monitoring MIB, V0.<u>8</u>71



468

469 3. System Configurations for the Job Monitoring MIB

470 This section enumerates the threetwo configurations for which the Job Monitoring MIB is intended to be used. The diagram in the Printer MIB entitled: "One Printer's View of the 471 Network"[1] is assumed for this MIB as well. Please refer to that diagram to aid in 472 473 understand the following system configurations. To simplify the pictures, the *devices* are

474 shown as printers. See Goals section.

475 3.1 Configuration 1 - client-printer

476 In the **client-printer** configuration, the **client**(s) submit jobs directly to the printer, either 477 by some direct connect, or by network connection. The **client-printer** configuration can accommodate multiple job submitting **clients** in either of two ways: 478

- 479 1. if each **client** relinquishes control of the Print Job Delivery Channel after each 480 job (or after a number of jobs)
- 481 2. if the printer supports more than one Print Job Delivery Channel

482 The job submitting **client** and/or **monitoring application** monitor jobs by

483 communicatinges directly with an agent that is part of the printer. The agent in the printer 484 shall keep the job in the Job Monitoring MIB as long as the job is in the Printer, and longer in order to implement the completed state in which monitoring programs can copy 485

486 out the accounting data from the Job Monitoring MIB.

487

488	all	end-user	########	SNMP query
489	++	++	job	submission
490	monitor	client	2	
491	++	+#++		
492	#	#		
493	# ######	######		
494	# #			
495	+==+===#=#=+==+			
496	agent			
497	++			
498	PRINTER <	+		
499		Print Job Del	ivery Chann	lel
500			-	
501	+==============+			

502 Figure 1 - Configuration 1 - client-printer - agent in the printer

503 The Job Monitoring MIB is designed to support the following relationships (not shown in 504 Figure 1):

- 505 1. Multiple clients may submit jobs to a printer.
- 506 2. Multiple clients may monitor a printer.
- 507 3. Multiple **monitors** may monitor a **printer**.

- 508 4. A **client** may submit jobs to multiple **printers**.
- 509 5. A monitor may monitor multiple printers.

510 **3.2** Configuration 2 - client-server-printer - agent in the server

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

512 server by some network connection, *not* directly to the **printer**. While configuration 2 is 513 included, the design center for this MIB is configurations 1 and 3,

The job submitting **client** and/or **monitoring application** monitor job by 514

- 515 communicatinges directly with:
- 516

517

1. a Job Monitoring MIB agent that is part of the server (or a front for the server)

518 There is no SNMP Job Monitoring MIB agent in the printer in configuration 2, at least 519 that the client or monitor are aware. In this configuration, the agent shall return the 520 current values of the objects in the Job Monitoring MIB both for jobs the server keeps and jobs that the server has submitted to the printer. In configuration 2, the server keeps a 521 522 copy of the job during the time that the server has submitted the job to the printer. Only 523 some time *after* the printer completes the job, shall the server remove the representation of the job from the Job Monitoring MIB in the server. The agent need not access the printer, 524 525 except when a monitor queries the agent using an SNMP Get for an object in the Job 526 Monitoring MIB. Or the agent can subscribe to the notification events that the printer 527 generates and keep the Job Monitoring MIB update to date. The agent in the server shall 528 keep the job in the Job Monitoring MIB as long as the job is in the Printer, and longer in 529 order to implement the **completed** state in which monitoring programs can copy out the 530 accounting data from the Job Monitoring MIB.



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

Job Monitoring MIB, V0.<u>8</u>71

- 555 The Job Monitoring MIB is designed to support the following relationships (not shown in 556 Figure 2):
- 557 1. Multiple **clients** may submit jobs to a **server**.
- 558 2. Multiple **clients** may monitor a **server**.
- 559 3. Multiple **monitors** may monitor a **server**.
- 560 4. A **client** may submit jobs to multiple **servers**.
- 561 5. A **monitor** may monitor multiple **servers**.
- 562 6. Multiple **servers** may submit jobs to a **printer**.
- 563 7. Multiple servers may control a printer.

564 **3.2** Configuration 3 - client-server-printer - client monitors printer agent and server

565 In the **client-server-printer** configuration 3, the **client**(s) submit jobs to an intermediate 566 **server** by some network connection, *not* directly to the **printer**.

567 The job submitting **client** and/or **monitoring application** <u>monitor jobs by</u>

- 568 communicatinges directly with:
- the server using <u>somea non-SNMP</u> protocol to monitor jobs in the server <u>that</u>
 <u>does not contain the Job Monitoring MIB</u> AND
- 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).

575 There is no SNMP Job Monitoring MIB agent in the server in configuration 3, at least that 576 the client or monitor are aware. In this configuration, the agent (in the printer) shall keep the values of the objects in the Job Monitoring MIB that the agent implements updated for 577 578 a job that the server has submitted to the printer. The agent shall obtain information about 579 the jobs submitted to the printer from the server (either in the job submission protocol, in 580 the document data, or by direct query of the server), in order to populate some of the objects the Job Monitoring MIB in the printer. The agent in the printer shall keep the job 581 582 in the Job Monitoring MIB as long as the job is in the Printer, and longer in order to 583 implement the **completed** state in which monitoring programs can copy out the 584 accounting data from the Job Monitoring MIB.



Job Monitoring MIB, V0.<u>871</u>

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

610 The Job Monitoring MIB is designed to support the following relationships (not shown in611 Figure 3):

- 612 1. Multiple **clients** may submit jobs to a **server**.
- 613 2. Multiple **clients** may monitor a **server**.
- 614 3. Multiple **monitors** may monitor a **server**.
- 615 4. A **client** may submit jobs to multiple **servers**.
- 616 5. A **monitor** may monitor multiple **servers**.
- 6. Multiple **servers** may submit jobs to a **printer**.
- 618 7. Multiple **servers** may control a **printer**.

4. Conformance Considerations 619

- 620 In order to achieve interoperability between job monitoring applications and job 621 monitoring agents, this specification includes the conformance requirements for both
- 622 monitoring applications and agents.

623 **3.2 Conformance Terminology**

- 624 This specification uses the verbs: "shall", "should", "may", and "need not" to specify conformance requirements as follows: 625
- 626 "shall": indicates an action that the subject of the sentence must implement in order • 627 to claim conformance to this specification
- 628 • "may": indicates an action that the subject of the sentence does not have to 629 implement in order to claim conformance to this specification, in other words that action is an implementation option 630
- 631 • "need not": indicates an action that the subject of the sentence does not have to 632 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. 633
- 634 • "should": indicates an action that is recommended for the subject of the sentence to 635 implement, but is not required, in order to claim conformance to this specification.

636 3.3 Agent Conformance Requirements

637 An agent shall implement all mandatory groups in this specification. An agent shall

- 638 implement conditionally mandatory groups, if the server or device that the agent is
- 639 instrumenting has the features represented by the objects in the conditionally mandatory
- 640 group. This section also lists the objects from other IETF MIB specifications that are
- mandatory for conformance by an agent to this Job Monitoring MIB specification. 641

642 3.3.1 MIB II System Group objects

643 The Job Monitoring MIB agent shall implement all objects in the system group of MIB-II 644 (RFC 1213), whether the Printer MIB is implemented or not.

645 3.3.2 MIB II Interface Group objects

- 646 The Job Monitoring MIB agent shall implement all objects in the Interfaces Group of
- 647 MIB-II (RFC 1213), whether the Printer MIB is implemented or not.

3.3.3 Printer MIB objects 648

- 649 If the agent is instrumenting a device that is a printer, the agent shall implement all of the
- 650 mandatory objects in the Printer MIB and all the objects in other MIBs that conformance
- 651 to the Printer MIB requires, such as the Host Resources MIB. If the agent is

652 instrumenting a server that controls one or more networked printers, the agent need not653 implement the Printer MIB and need not implement the Host Resources MIB.

mplement the rinker wild and need not implement the riost Resources will

654 **3.4 Job Monitoring Application Conformance Requirements**

A job monitoring application (monitor) is a management or client application that uses
 SNMP to access the agent that implements this Job Monitoring MIB. A job monitoring

657 application shall accept all objects in all mandatory and conditionally mandatory groups

that are required to be implemented by an agent according to Section 3.3 and shall either

659 present them to the user or ignore them.

A job monitoring application shall accept all enum values and bit vector bits specified in
this standard and additional ones that may be registered with IANA and shall either
present them to the user or ignore them. See Section 6 entitled "IANA Considerations"
on page 28.

664 **4. Job Identification**

665 There are a number of attributes that permit a user, operator or system administrator to identify jobs of interest, such as jobOwner, jobName, etc. In addition, there is a The 666 purpose of the Job Submission ID object that allows Identification objecta s is to allow the 667 monitoring application user, operator, or the system administrator to quickly locate and 668 669 identify a particular the jobs of interest that was submitted from a particular client by the 670 user invoking the monitoring application. The Job Monitoring MIB needs to provide for identification of the job at both sides of the job submission process. The primary 671 672 identification point is must be at the client side. The Job Submission ID client side 673 identifiers allows the monitoring application user to identify the job of interest from all the 674 jobs currently "known" by the server or device. The Job Submission IDelient side 675 identifiers can be assigned by either the client's local system or a downstream server or 676 device. The point of assignment will be determined by the job submission protocol in use. Two client-side objects are provided: jmJobIdName and jmJobIdNumber so that both 677 678 textual identifiers and numeric identifiers can be represented, depending on the job 679 submission protocol. The intent is that the agent shall provide the same values for these 680 two client-side objects as the user is provided for by the job submission protocol that 681 happens to be in use. The client side job identifiers in combination should provide the user 682 and operator with unique job identifications. 683 The server/device-side identifier, called the **jmJobIndex** object, will be assigned by the

684 server or device that accepts the jobs from submitting clients. The MIB agent shall use

685 the job identifier assigned by the server or device to the job as the value of the 686 **jmJobIndex** object that defines the table rows (there are multiple tables) that contain the

- information relating to the job. This object allows the interested party to obtain all objects
- 688 desired that relate to this job. The MIB provides a mapping table that maps each Job
- 689 Submission ID to the corresponding **jmJobIndex** value, so that an application can
- 690 determine the correct value for the jmJobIndex value for the job of interest in a single Get
- 691 operation. See the **jmJobIDGroup** on page 77.

The jmJobName <u>attributeobject</u> provides a name that the user supplies an a job attribute
 with the job. It is not necessarily unique, even for one user, let alone across users.

694 **5. Internationalization Considerations**

695 There are a number of objects in this MIB that are represented as coded character sets. 696 The data type for such objects is **OCTET STRING**. Such objects could be in different 697 coded character sets and could be localized in the language and country, i.e., could be 698 localized. However, for the Job Monitoring MIB, most of the objects are supplied as job 699 attributes by the client that submits the job to the server or device and so are represented 700 in the coded character set specified by that client. Therefore, the agent is not able to 701 provide for different representations depending on the locale of the server, device, or user 702 of the job monitoring application. The only exception is job submission protocols that 703 pass job or document attributes as OBJECT IDENTIFIERS or enums. For those job and 704 document attributes, the agent shall represent the corresponding objects in the Job 705 Monitoring MIB as coded character sets in the current (default) locale of the server or 706 printer as established by the system administrator or the implementation.

For simplicity, this specification assumes that the clients, job monitoring applications,

servers, and devices are all running in the same locale. However, this specification allows

them to run in any locale, including locales that use two-octet coded character sets, such

as ISO 10646 (Unicode). Job monitors applications are expected to understand the coded

character set of the client (and job), server, or device. No special means is provided for

the monitor to discover the coded character set used by jobs or by the server or device.

713 This specification does *not* contain an object that indicates what locale the server or device

is running in, let alone contain an object to control what locale the agent is to use to

715 represent coded character set objects.

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

convention from SNMPv2-TC (RFC 1903). The job management application shall display
 such objects in the locale of the user running the monitoring application.

719 6. IANA Considerations

720 During the development of this standard, the Printer Working Group (PWG) working with

721 IANA will register additional enums and bit strings 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 and bit strings after this standard is approved in
- cooperation with an IANA-appointed registration editor from the PWG according to the
- 725 procedures described in this section:

726 6.1 IANA Registration of enums

- 727 This specification uses textual conventions to define enumerated values (enums).
- Enumerations (enums) are sets of symbolic values defined for use with one or more
- 729 objects. All enumeration 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 listed at the beginning of the MIB module specification.
- This working group has defined several type of enumerations for use in the Job
- 733 Monitoring MIB and the Printer MIB (see RFC 1759). These enumerations differ in the
- method 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. The definitions of these types of enumerations are:
- 737 Type 1 enumeration: All the values are defined in the Job Monitoring MIB specification
- 738 (RFC for the Job Monitoring MIB). Additional enumerated values require a new RFC.
- 739 NOTE There are no type 1 enums in the current draft.
- 740 Type 2 enumeration: An initial set of values are defined in the Job Monitoring MIB
- specification. Additional enumerated values are registered after review by this working
- rd2 group. The initial versions of the MIB will contain the values registered so far. After the
- 743 MIB is approved, additional values will be registered through IANA after approval by this
- 744 working group.
- The following type 2 enums are contained in the current draft (see table of contents Tableof Textual-Conventions):
- 747 **1. JmJobServiceTypesTC**
- 748 **2. JmJobStateTC**
- 749 3. JmAttributeTypeTC
- 750 Type 3 enumeration: An initial set of values are defined in the Job Monitoring MIB
- 751 specification. Additional enumerated values are registered without working group review.
- 752 The initial versions of the MIB will contain the values registered so far. After the MIB is
- approved, additional values will be registered through IANA without approval by this
- working group.
- 755 NOTE There are no type 3 enums in the current draft.

756 **6.2 IANA Registration of bit string values**

- 757 This draft contains the following bit string textual-conventions:
- 758 1. JmJobStateReasonsTC
- 759 The **jobStateReasons** attribute is defined as a bit string using the
- 760 **JmJobStateReasonsTC** textual-convention that is represented by an **OCTET**
- 761 **STRING(SIZE(0..63)**). Bits in the bit string are assigned starting with the most
- significant bit in the most significant octet which is called bit 1. Bit 2 is the next most
- significant bit in the most significant octet, etc. Bit 9 is the most significant bit in the
- second most significant octet, etc., up to the maximum bit: $504 (= 8 \times 63)$. The
- registration of **JmJobStateReasonsTC** bit values shall follow the procedures for a type 2
- enum as specified in Section 6.1

767 **7. Security Considerations**

768 **7.1 Read-Write objects**

All objects are read-only greatly simplifying the security considerations. If another MIB augments this MIB, that MIB might allow objects in this MIB to be modified. However, that MIB shall have to support the required access control in order to achieve security, not this MIB.

773 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

776 jobKOctetsRequestedTotal and jobKOctetsCompleted attributes, so that a user can

tell how busy a printer is. Other sites might allow all unprivileged users to see all objects

of all jobs. It is up to the agent to implement any such restrictions based on the

- identification of the user making the SNMP request. This MIB does not require, nor does
- 780 it specify how, such restrictions would be implemented. A monitoring application should

781 <u>enforce the site security policy with respect to returning information to an unprivileged</u>

782 end user that is using the monitoring application to monitor jobs that do not belong to that

- 783 <u>user, i.e., the jobOwner attribute in the jmAttributeTable does not match the user's user</u>
- 784 <u>name. See the **JmAttributeTypeTC** textual convention on page 50 and the</u>

785 jmAttributeTable.

An operator is a privileged user that would be able to see all objects of all jobs,

787 independent of the policy for unprivileged users.

788 **8. Returning Objects With No Value In Mandatory Groups**

If an object in a mandatory group does not have an instrumented value for a particular job submission protocol or the job submitting client did not supply a value (and the accepting server or device does not supply a default), this MIB requires that the agent shall follow the normal SNMP practice of returning a distinguished value, such as a zero-length string, a **unknown(2)** for an enum, or a (-2) for an integer value.

794 **9. Notification and Traps**

795 This MIB does not specify any traps. For simplicity, management applications are

resulting network traffic is not expected to be significant.

797 **10. MIB specification**

The following pages constitute the actual Job Monitoring MIB.

Job Monitoring MIB, V0.871 April 4, 1997

```
799
     Job-Monitoring-MIB DEFINITIONS ::= BEGIN
800
801
     IMPORTS
        MODULE-IDENTITY, OBJECT-TYPE, experimental,
        Integer32
                                                           FROM SNMPv2-SMI
        TEXTUAL-CONVENTION, DateAndTime
                                                           FROM SNMPv2-TC
        MODULE-COMPLIANCE, OBJECT-GROUP
                                                           FROM SNMPv2-CONF;
        -- The following textual-conventions are needed
        -- to implement certain attributes, but are not
        -- needed to compile this MIB. They are
        -- provided here for convenience:
        -- DateAndTime
                                                           FROM SNMPv2-TC
        -- PrtAlertCodeTC, PrtInterpreterLangFamilyTC
                                                           FROM Printer-MIB
802
803
     -- Use the experimental (54) OID assigned to the Printer MIB before it
804
     -- was published as RFC 1759.
805
     -- Upon publication of the Job Monitoring MIB as an RFC, delete this
     -- comment and the line following this comment and change the
806
     -- reference of { temp 104 } (below) to { mib-2 X }.
807
808
     -- This will result in changing:
809
     -- 1 3 6 1 3 54 jobmonmib(105)
                                        to:
810
     --13612 1 jobmonmib(X)
811
     -- This will make it easier to translate prototypes to
812
     -- the standard namespace because the lengths of the OIDs won't
813
     -- change.
814
     temp OBJECT IDENTIFIER ::= { experimental 54 }
815
816
     jobmonmib MODULE-IDENTITY
817
         LAST-UPDATED "97040403140000Z"
818
         ORGANIZATION "IETF Printer MIB Working Group"
819
         CONTACT-INFO
820
             "Tom Hastings
821
             Postal:
                      Xerox Corp.
822
                      Mail stop ESAE-231
823
                      701 S. Aviation Blvd.
824
                      El Segundo, CA 90245
825
826
                      (301)333-6413
             Tel:
             Fax:
                      (301)333-5514
827
828
             E-mail: hastings@cp10.es.xerox.com"
829
         DESCRIPTION
830
             "The MIB module for monitoring job in servers, printers, and
831
             other devices.
832
833
             File: jmp-mib.doc, .pdf, .txt, .mib
834
             Version: 0.871"
835
         ::= { temp 105 }
836
```

338	- Tortual conventions for this MIP module
3 <i>39</i> 340	Textual conventions for this MiB module
R/1	
2/2	
342 843	STATUS current
344	DESCRIPTION
345	"The simple time at which an event took place. The units are in
346	seconds since the system was booted.
347	NOWN THE ACTION OF A STREET AND A STREET AS A STREET AND A STREET AND A STREET AS A STREET
548 270	NOTE - JMTIMETC is defined in units of seconds, rather than
349	(even if they have to implement the 100ths of a second to comply
351	with MIB-II.)
352	
353	NOTE - JmTimeTC is defined as an Integer32 so that it can be
354	used as a value of an attribute, i.e., as a value of the
333 954	jmAttributeValueAsInteger object (see page 90). The TimeStamp
850 857	3 IMPLICIT INTEGED tag not an Integer32 so cannot be used in
858	this MIB as one of the values of imAttributeValueAsInteger ."
359	SYNTAX INTEGER(02147483647)
860	
361	
362	
363	
364	JmTimeIntervalTC ::= TEXTUAL-CONVENTION
864 865	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current
364 365 366	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION
364 365 366 367 368	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds.
864 865 866 867 868 868	<u>JmTimeIntervalTC</u> ::= TEXTUAL-CONVENTION <u>STATUS</u> current <u>DESCRIPTION</u> <u>"A period of time, measured in units of seconds.</u> NOTE - JmTimeIntervalTC is defined in the same units as
64 65 66 67 68 69 70	<u>JmTimeIntervalTC ::= TEXTUAL-CONVENTION</u> <u>STATUS current</u> <u>DESCRIPTION</u> <u>"A period of time, measured in units of seconds.</u> <u>NOTE - JmTimeIntervalTC is defined in the same units as</u> <u>JmTimeTC, namely seconds.</u>
364 365 366 367 368 369 370 371	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds.
364 365 366 367 368 369 370 371 372	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it
364 365 366 367 368 369 370 371 372 373	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as
664 665 666 667 668 669 670 671 672 673 674 674	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90).
64 65 66 67 68 69 70 71 72 73 74 75 76	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is
364 365 366 367 368 369 370 371 372 373 374 375 376 376 377	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second not in
364 365 366 367 368 369 370 371 372 373 374 375 376 377 378	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second, not in units of seconds."
664 665 666 667 668 669 670 671 672 673 674 675 674 675 677 678 679	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second, not in units of seconds." SYNTAX
364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second, not in units of seconds." SYNTAX
364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second, not in units of seconds." SYNTAX
364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second, not in units of seconds." SYNTAX
364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second, not in units of seconds." SYNTAX
364 365 366 367 368 369 370 371 372 373 374 375 376 377 380 381 382 383 384	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second, not in units of seconds." SYNTAX INTEGER(02147483647)
364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 380 381 382 383 384 385	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second, not in units of seconds." SYNTAX INTEGER(02147483647) JmJobStateTC ::= TEXTUAL-CONVENTION STATUS current
364 365 366 367 368 369 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386	JmTimeIntervalTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION "A period of time, measured in units of seconds. NOTE - JmTimeIntervalTC is defined in the same units as JmTimeTC, namely seconds. NOTE - JmTimeIntervalTC is defined as an Integer32 so that it can be used as a value of an attribute which is represented as the value of the jmAttributeValueAsInteger object (see page 90). The TimeIntervalTC textual-convention defined in SNMP-TC is defined as an Integer32, so it could be used in this MIB, except that TimeIntervalTC is defined in 100ths of a second, not in units of seconds." SYNTAX INTEGER(02147483647) JmJobStateTC ::= TEXTUAL-CONVENTION STATUS current DESCRIPTION

888				
889	Management applications shall be prepared to receive all the			
890	standard job states. Servers and devices are not required to			
891	generate all job states, only those which are appropriate for			
892	the particular implementation However the following states			
803	are mandatory for a gerver or device implementation:			
895 804	are mandatory for a server of device imprementation.			
094 005				
893	processing(5)			
896	needsAttention(7)			
897	canceled(8)			
898	<pre>completed(9)</pre>			
899				
900	<u>See Section 11 entitled 'Job Life Cycle' on page 95 for</u>			
901	additional job state semantics, legal job state transitions, and			
902	implementation considerations.			
903				
904	A companion textual convention (JmJobStateReasonsTC) and			
905	corresponding attribute (jobStateReasons) provide additional			
906	information about job states. While the job states cannot be			
907	added to without impacting deployed clients it is the intent			
908	that additional .Tm.TobStateReasonsTC enums can be defined without			
000	impacting deployed alignta In other words the			
010	Impacting deproyed citents. In other words, the			
910	JEDODStateReasonsic is intended to be extensible. See page 56.			
911				
912	The following job state standard values are defined by adding			
913	(+2) to the last arc of the ISO DPA OBJECT IDENTIFIER Value of			
914	the Job-Current-State Job attribute:"			
915				
916	This is a type 2 enumeration. See Section 6.1 on page 28.			
917	SYNTAX INTEGER {			
	other(1), The job state is not one of the defined			
	states.			
	unknown(2), The job state is not known, or is			
	indeterminate.			
	preProcessing(3), - The job has been created on the server or			
	device but the submitting client is in			
	- processing. The job maybe in the process			
	of being checked by the server/device for			
	attributes, defaults being applied, a			
	- device being selected etc.			
I				
I	held (312) The job is not yet a candidate for			
I	$= \operatorname{processing for any number of reasons}$			
	= The reasons are represented as hits in			
	the debdteteDeegeng attribute			
	the jubstatereasons attribute. Some			
	reasons are used in other states to give			
	added information about the job state.			
	See the JmJobStateReasonsTC textual			
	convention for the specification of each			

	reason and in which states the reasons may be used.
<pre>pending(46),</pre>	The job is a candidate for processing, but is not yet processing.
processing(57),	 The job is using one or more document transforms which include purely software processes, such as interpreting a PDL, and hardware devices, but is not yet making marks on a medium. If an implementation does not distinguish between processing and printing, then the processing state shall be implemented.
<pre>printing(6)</pre>	 The job is printing, i.e., making marks on a medium. If an implementation does not distinguish between processing and printing, then the processing state shall be implemented.
needsAttention(7 9) ,	 The job is using one or more devices, but has encountered a problem with at least one device that requires human intervention before the job can continue using that device. Examples include running out of paper or a paper jam. Usually devices indicate their condition
	 in human readable form locally at the device. The management application can obtain more complete device status remotely by querying the appropriate device MIB using the job's jmDeviceIndex object in the Job Monitoring MIB.
	NOTE Instead of the needsAttention job state, ISO DPA uses the multi valued printer-state-of-printers-assigned job attribute, so that the state of each device that a job is using can be accurately represented. However, for the Job Monitoring MIB, the simpler approach is used of adding a single needsAttention job state if any device that the job is using needs attention and relying on the device MIB for more information.
paused(13),	 The job has been indefinitely suspended by a client issuing an operation to suspend the job so that other jobs may proceed using the same devices. The client may issue an operation to resume

Job Monitoring MIB, V0.<u>8</u>71

	 the paused job at any time, in which case the server or printer places the job in the held or pending states and the job is eventually resumed at the point where the job was paused.
interrupted(8),	 The job has been interrupted while processing by a client issuing an operation that specifies another job to be run instead of the current job. The server or printer will automatically resume the interrupted job when the interrupting job completes.
<pre>canceledterminatin g(814),</pre>	 The job is in the process of being terminated by the server or device or has completed terminating the jobprinter, either because the client canceled the job or because a serious problem was encountered by a document transform while processing the job. The job's jobStateReasons attribute shall contain the reasons that the job was canceledterminated. The job shall remain in the canceled state for the same period of time as if the job had completed, before transiting to the unknown state. See the completed state description.
<pre>retained(11),</pre>	<pre>— The job is being retained by the server or printer after processing and all of the media have been successfully stacked in the output bin(s).</pre>

```
— The retained state is conditionally
                   - mandatory. Implementations that do not
                   - retain jobs after they are finished
                   - processing such that the client could
                   - request that the job be repeated (or
                   - resubmitted), need not implement the
                   - retained state.
completed(917)
                   -- The job has (1) completed after
                   -- processing/printing and all of the media
                   -- have been successfully stacked in the
                    -- output bin(s) and (2) the server/device
                    -- is keeping the job in summary form for a
                    -- site-settable period for purposes of
                    -- aiding operators and users to determine
                    -- the disposition of users' jobs.
                    _ _
                    -- The job (1) has completed successfully or
                    -- with warnings or errors, (2) has been
                    -- aborted while printing by the
                    -- server/device, or (3) has been cancelled
                    -- by the submitting user or operator before
                    -- <del>or during processing</del>.
                                              The job's
                    -- jobStateReasons attributeobject shall
                    -- contain the reasons that the job has
                    -- entered the completed state.
                    -- While in the completed state, a job's
                    -- document data (and submitted resources,
                    -- such as fonts, logos, and forms, if any)
                    -- need not be retained by the server; thus
                    -- a job in the completed state could not be
                    -- reprinted. The length of time that a job
                    -- may be in the completedthis state, before
                    -- transitioning to unknown, is specified by
                    -- the value of the jmGeneralJobPersistence
                    -- objectimplementation dependent.
                                                        Tn
                    -- addition, the agent shall maintain all of
                   -- the attributes in the jmAttributeTable
                    -- for at least the time specified in the
                    -- jmGeneralAttributePersistence object,
                    -- However, servers that implement the
                       completed job state shall retain all of
                       the job's Job Monitoring MIB objects,
                       except the jmQueueGroup objects, so that
                       a management application accounting
                       program can copy all the attributes them
                       to an accounting log.
```

918

}
920 921 JmAttributeTypeTC ::= TEXTUAL-CONVENTION 922 current STATUS 923 DESCRIPTION 924 "The type of the attribute. 925 926 Some attributes represent information about a job, such as a 927 file-name, or a document-name, or submission-time or completion 928 time. Other attributes represent resources required, e.q., a 929 medium or a colorant , etc. to process the job before the job 930 start processing OR to indicate the amount of the resource that 931 is being consumed while the job is processing, e.g., pages 932 completed or impressions completed. If both a required and a consumed value of a resource is needed, this specification 933 934 assigns two separate attribute enums are assigned in the textual 935 convention. 936 937 Most attributes items shall have only one row per job. However, 938 a few attributes items can have multiple values per job or even 939 per document, where each value is a separate row in the 940 jmAttributeTable. Unless indicated otherwise in 941 JmAttributeTypeTC, an agent shall ensure that each attribute item occurs only once in the **jmAttributeTable** for a job. 942 Attributes items that may appear multiple times in the 943 944 jmAttributeTable for a job are indicated in their specification 945 in the JmAttributeTypeTC (see page 37). However, such attribute 946 items shall not contain duplicates for 'intensive' (as opposed 947 to 'extensive') attributes. 948 949 For example, each **documentFormatEnum** attribute entry 950 shall appear in the **jmAttributeTable** only once for a job 951 since the interpreter language is an intensive attribute 952 item, even though the job has a number of documents that 953 all use the same PDL. 954 955 As another example of an intensive attribute that can 956 have multiple entries, if a document or job uses multiple types of media, there shall be only one row in the **jmAttributeTable** for each media type, not one row 957 958 959 for each document that uses that medium type. 960 On the other hand, if a job contains two documents of 961 the same name, there can be separate rows for the 962 963 documentName attribute item with the same name, since a 964 document name is an extensive attribute item. 965 966 In the following definitions of the enums, each description 967 indicates whether the value of the attribute shall be 968 represented using the jmAttributeValueAsInteger or the 969 jmAttributeValueAsOctets objects by the initial tag: 'Integer:' 970 or 'Octets:', respectively. A very few attributes use both

919

[Page 37]

971 972 973	objects at the same time to represent a pair of values (mediumConsumed) and so have both tags. See the imattributeGroup for the descriptions of these objects
974	JARCEI Ducedi oup for the descriptions of these objects.
975	If the imAttributeValueAsInteger object is not used (no
976	'Integer: ' tag), the agent shall return the value (-1)
977	indicating other. If the imAttributeValueAsOctets object is not
978	used (no 'Octets:' tag), the agent shall return a zero-length
979	octet string.
980	
981	An agent shall create a row in the imAttributeTable for each
982	attribute that is (1) supplied with a job when the job is
983	accepted by a server or printer or that (2) the server or
984	printer supplies as a default either when the job is accepted or
985	later during processing. An agent shall not create a row for
986	any attribute that was neither supplied with the job nor
987	supplied by the server or printer as a default.
988	
989	Some attributes are mandatory for conformance, and the rest are
990	conditionally mandatory. An agent shall instrument any
991	mandatory attribute. If the server or printer does not provide
992	access to the information about the mandatory attribute, the
993	agent shall return the 'unknown' value. An agent shall
994	instrument any conditionally mandatory attribute if the server
995	or printer provides access to the information about the
996	attribute to the agent. If the server or printer does not
997	provide access to the information about the conditionally
998	mandatory attribute, the agent shall not create the row in the
999	imAttributeTable.
1000	
1001	The mandatory attributes are the ones required to have copies in
1002	the jmJobStateTable . The mandatory attributes are:
1003	
1004	jobState
1005	numberOfInterveningJobs
1006	deviceAlertCode
1007	jobKOctetsRequestedTotal
1008	jobKOctetsCompleted
1009	impressionsRequestedTotal
1010	impressionsCompleted
1011	outputBinName
1012	
1013	The table of contents lists the attributes in order to help see
1014	the order of OID assignment which is the order that the GetNext
1015	operation returns attributes.
1016	
1017	The standard attribute types defined so far are:"
1018	
1019	This is a type 2 enumeration. See Section 6.1 on page 28.
1020 SYN	VTAX INTEGER {
	- jm Description - including Octets: or Integer:
	- Attribute to specify whether the value is represented
	- TypeIndex in the jmAttributeValueAsOctets or the

```
jmAttributeValueAsInteger object,
               respectively.
_ _
other(1),
            -- An attribute that is not in the list and/or
            -- that has not been registered with IANA.
-- Job State attributes
_ _
-- The following attributes specify the state of a job.
jobState(2)
            -- The current state of the job (pending,
            -- processing, held, etc.)
            _ _
               Management applications shall be prepared to
               receive all the standard job states.
               Servers and devices are not required to
               generate all job states, only those which
               are appropriate for the particular
               implementation.
               A companion textual convention
               (JmJobStateReasonsTC) and corresponding
               attribute (jobStateReasons) provide
               additional information about job states.
               While the job states cannot be added to
               without impacting deployed clients, it is
               the intent that additional
               JmJobStateReasonsTC enums can be defined
               without impacting deployed clients. In
               other words, the JmJobStateReasonsTC is
               intended to be extensible. See page 56.
               This attribute is a type 2 enum.
jobStateAsso -- Integer: The value of the most relevant
ciatedValue( -- attribute associated with the job's current
3)
            -- state.
               Which attribute depends on the job's current
               state (as specified by the value of the
               jmJobState object and the jobState
               attribute) as follows:
               jmJobState
                               Associated Attribute
                                                    Page
               /jobState
                               jobStartedBeingHeldTime
               held
                                                      53
               pending
                               numberOfInterveningJobs
                                                      43
               processing
                               jobKOctetsRequested
                                                      47
               printing
                               impressionsRequested
                                                      49
               needsAttention
                               deviceAlertCode
                                                      41
```

50

46

canceled impressionsCompleted completed outputBinName

NOTE - The **jobStateAssociatedValue** attribute selects from amongst seven mandatory attributes that attribute that is most relevant to the job's current state. the **jobStateAssociatedValue** attribute is provided as an efficiency improvement, so that an application can obtain the most relevant attribute for each job's current state (1) without first having to determine the job's state or (2) having to request all seven mandatory attributes in the same GetNext operation that obtains the next job in the next conceptual row in the **jmAttributeTable**.

-- Octets: Additional information regarding jobStateReas -- the **jmJobState**/**jobState** object/attribute. ons(4) -- The jobStateReasons attribute identifies the reason or reasons that the job is in the held, pending, processing, needsAttention, canceled, retained, or completed state. The server shall indicate the particular reason(s) by setting the value of the jobStateReasons attribute. While the job states cannot be added to without impacting deployed clients, it is the intent that additional JmJobStateReasonsTC enums can be defined without impacting deployed clients. In other words, the JmJobStateReasonsTC is intended to be extensible. See page 56. When the job does not have any reasons for

being in its current state, the server shall set the value of the **jobStateReasons** attribute to a bit string containing all zeros.

Bits in the bit string are assigned starting with the most significant bit in the most significant octet which is called bit 1. Bit 2 is the next most significant bit in the most significant octet, etc. Bit 9 is the most significant bit in the second most significant octet, etc., up to the maximum bit: 504 (= 8 x 63). See JobStateReasonsTC on page 56.

An agent only needs to return the most significant octet up to the least significant octet that contains a non-zero

Bergman, Hastings, Isaacson, Lewis

[Page 40]

bit. The remaining octets need not be returned.

If all bits are zero, the agent may return an OCTET STRING of zero length. Alternatively, an agent may always return a fixed number of octets starting with the most significant octet and running through the least significant octet that could ever have a one bit in it for that implementation.

This object is a type 2 bit string. See Section 6 entitled 'IANA Considerations' on page 28.

numberOfInte rveningJobs(5) Integer: The number of jobs that are expected to be processed before this job is processed according to the implementation's queuing algorithm if no other jobs were to be submitted. In other words, this value is the job's queue position. The agent shall return a value of 0 for this object when the job starts processing (since there are no jobs in front of the job).

deviceAlertC -- The device alert code when the job is
ode(6) -- stopped because the device needs attention ,
 -- i.e., needs human intervention. When the
 -- device is a printer, this device alert code
 -- shall be the printer alert code defined by
 -- the Printer MIB using the PrtAlertCodeTC
 -- textual convention or equivalent.

<pre>processingMe ssage(7),</pre>	 Octets: A coded character set message that is generated during the processing of the
	 job as a simple form of processing log to
	 show progress and any problems.
	 A row with this attribute item may appear
	 more than once in the jmAttributeTable for a
	 job.

	<pre> to help the user distinguish between his/her various jobs. This name does not need to be unique.</pre>
	 This attribute is intended for enabling a user or the user's application to convey a job name that may be printed on a start sheet, returned in a query result, or used in notification or logging messages.
	If this attribute is not specified when the job is submitted, no job name is assumed, but implementation specific defaults are allowed, such as the value of the documentName(4) resource item of the first document in the job or the fileName(3) resource item of the first document in the job.
	The jobName attribute is distinguished from the jobComment attribute, in that the jobName attribute is intended to permit the submitting user to distinguish between different jobs that he/she has submitted. The jobComment attribute is intended to be free form additional information that a user might wish to use to communicate with himself/herself, such as a reminder of what to do with the results or to indicate a different set of input parameters were tried in several different job submissions.
jobServiceTy pes(9)	Integer: Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.) as defined by the JmJobServiceTypesTC on page 54. The service type is represented as a BITS datatype 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, fax, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the values: 8+32+4=44, respectively.
	 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. This attribute

[Page 42]

	<pre> shall be implemented if the server or device has other types in addition to or instead of printing.</pre>
	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 object is in the job identification category.
	This attribute is a type 2 enum.
jobOwner(10)	Octets: The coded character set name of the user that submitted the job. The method of assigning this user name will be system and/or site specific but the method must insure that the name is unique to the network that is visible to the client and target device.
	 This value should be the <i>authenticated</i> name of the user submitting the job.
jobAccountNa me(<mark>115</mark>),	 Octets: Arbitrary binary information which may be coded character set data or encrypted data supplied by the submitting user for use by accounting services to allocate or categorize charges for services provided, such as a customer account name.
	NOTE: This attribute need not be printable characters.
jmJobDeviceN ameOrQueueRe quested(12)	The administratively defined coded character set name of the target device or queue. Its value corresponds to the Printer MIB: prtGeneralPrinterAdminName object (added to the draft Printer MIB) for printers. For servers, this object is the name that users supply to indicate whether they want the job to be processed, typically, but not limited to, a job queue name or logical printer name.
jobSourceCha nnelIndex(13 8),	Integer: The index of the row in the associated Printer MIB of the channel which is the source of the print job. See RFC 1759.
	Must be 1 or greater.
	NOTE - the Job Monitoring MIB points to the

I

	 Channel row in the Printer MIB, so there is no need for a port object in the Job Monitoring MIB, since the PWG is adding a prtChannelInformation object to the Channel table of the draft Printer MIB.
physicalDevi ceIndex(14),	Integer: The index of the physical device MIB instance requested/used, such as the Printer MIB. This value is an hrDeviceIndex value. See the Host Resource MIB.
	A row with this attribute item may appear more than once in the jmAttributeTable for a job that is using more than one physical device, but the jmAttributeValueAsInteger shall be different for each such row.
	 If there is no physical device MIB instance for this job, this row shall not be present in the jmAttributeTable.
physicalDevi ceName(15),	Octets: The name of the physical device to which the job is assigned.
	A row with this attribute item may appear More than once in the jmAttributeTable for a job that is using more than one physical device, but the jmAttributeValueAsOctets shall be different for each such row.
fileName(163),	Octets: The coded character set file name of the document.
	 A row with this attribute item may appear more than once in the jmAttributeTable for a job.
<pre>documentName (174),</pre>	Octets: The coded character set name of the document.
	 A row with this attribute item may appear more than once in the jmAttributeTable for a job.
jobComment(1 8 6),	 Octets: An arbitrary human-readable coded character text string supplied by the submitting user or the job submitting application program for any purpose. For example, a user might indicate what he/she is going to do with the printed output or the job submitting application program might indicate how the document was produced.

Bergman, Hastings, Isaacson, Lewis

[Page 44]

-- The **jobComment** attribute is not intended to -- be a name; see the jmJobName attributeobject. -- Job Parameter attributes ___ -- The following attributes represent input parameters -- supplied by the submitting client in the job submission -- protocol. jobPriority(Integer32(0..100): The priority for 19) scheduling the job. It is used by servers and devices that employ a priority-based scheduling algorithm. A higher value specifies a higher priority. The value 1 is defined to indicate the lowest possible priority (a job which a priority-based scheduling algorithm shall pass over in favor of higher priority jobs). The value 100 is defined to indicate the highest possible priority. Priority is expected to be evenly or 'normally' distributed across this range. The mapping of vendor-defined priority over this range is implementation-specific. A value of **0** shall be returned by implementations that do not have a prioritybased queuing algorithm. jobProcessAf -- Integer: The calendar date and time of day terDateAndTi -- after which the job shall become a candidate me(20) -- to be scheduled for processing. If the -- value of this attribute is in the future, -- the server shall set the value of the job's -- jmJobCurrentState object and the job's -- jobState attribute to held and add the -- jobProcessAfterSpecified bit value to the -- job's jobStateReasons attribute and shall -- not schedule the job for processing until -- the specified date and time has passed. -- When the specified date and time arrives, -- the server shall remove the -- jobProcessAfterSpecified bit value from the -- job's jobStateReasons attribute and, if no -- other reasons remain, shall change the job's -- jmJobCurrentState and the job's jobState -- attribute to **pending** so that the job becomes -- a candidate for being scheduled on

Bergman, Hastings, Isaacson, Lewis

[Page 45]

	devices(s).
	 The server shall assign an empty value to the jobProcessAfterDateAndTime attribute when no process after time has been specified, so that the job shall be a candidate for processing immediately.
outputBinInd ex(<mark>219</mark>),	 Integer: The output subunit index in the Printer MIB of the output bin to which all or part of the job is placed in.
	 A row with this attribute item may appear more than once in the jmAttributeTable for a job, but the jmAttributeValueAsInteger shall be different for each such row.
outputBinNam e(<mark>2210),</mark>	Octets: The name of the output bin to which all or part of the job is placed in.
	 A row with this attribute item may appear more than once in the jmAttributeTable for a job, but the jmAttributeValueAsOctets shall be different for each such row.
<pre>sides(2311),</pre>	 Integer: The number of sides that any document in this job will require or did use.
<pre>documentForm atIndex(2412),</pre>	Integer: The interpreter language family index in the Printer MIB of the prtInterpreterLangFamily object, that this job requires and uses. A document or a job may use more than one PDL. A row with this attribute item may appear more than once in the jmAttributeTable for a job, but the jmAttributeValueAsInteger shall be different for each such row. As with all intensive attribute items where multiple rows are allowed, there shall be only one distinct row for each distinct PDL; there shall be no duplicates. NOTE - This attribute type is intended to be used with an agent that implements the Printer MIB and shall not be used if the agent does not implement the Printer MIB.
documentForm atEnum(<mark>2513)</mark>	Integer: The interpreter language family corresponding to the Printer MIB

[Page 46]

-- prtInterpreterLangFamily object, that this -- job requires and uses. A document or a job -- may use more than one PDL. _ _ -- A row with this attribute item may appear -- more than once in the **jmAttributeTable** for a -- job, but the **jmAttributeValueAsInteger** shall -- be different for each such row. As with all -- intensive attribute items where multiple -- rows are allowed, there shall be only one -- distinct row for each distinct PDL; there -- shall be no duplicates. _ _ -- This enum is a type 2 enum. ___ -- NOTE: Theis PrtInterpreterLangFamilyTC -- textual convention is defined inimported -- from the draft Printer MIB, but is not in RFC 1759. -- Resources attributes (requested and consumed) attributes -- Pairs of these attributes can be used by monitoring -- applications to show users 'thermometers' of usage to users. jobCopiesReq -- Integer: The number of copies of the entire **uested**(2616) -- job that are to be produce _ _ -- A value of -2 means unknown. jobCopiesCom -- Integer: The number of copies of the entire pleted(2717) -- job that the entire job has completed so -- far. _ _ -- A value of (-2) means unknown. documentCopi -- Integer: The total count of the number of **esRequested(** -- document copies requested. If there are 2818), -- 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. documentCopi -- Integer: The total count of the number of esCompleted(-- document copies completed so far for the job -- as a whole. If there are documents A, B, 2919), -- 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.

Bergman, Hastings, Isaacson, Lewis

,

_ _

,

[Page 47]

The total number of K (1024) jobKOctetsRe -- Integer: questedTotal -- octets being requested to be processed in (3020), -- the job, including document and job copies. -- The agent shall round the actual number of -- octets up to the next highest K. Thus 0 -- octets shall be represented as 0, 1-1024 -- octets shall be represented as 1, 1025-2048 -- shall be represented as 2, etc. _ _ -- The server/device may update the value of -- this attribute after each document has been -- transferred to the server/device or the -- server/device may provide this value after -- all documents have been transferred to the -- server/device, depending on implementation. -- In other words, while the job is in the -- preProcessing state and when the job is in -- the **held** state with the **jmJjobStateReasons** -- attributeobject containing a documentsNeeded -- or **preProcessing** value, the value of the -- jobKOctetsRequestedTotal attribute depends -- on implementation and may not correctly -- reflect the size of the job. _ _ -- In computing this value, the server/device -- shall include the multiplicative factors -- contributed by (1) the number of document -- copies, and (2) the number of job copies, -- independent of whether the device can -- process multiple copies of the job or -- document without making multiple passes over -- the job or document data and independent of -- whether the output is collated or not. Thus -- the server/device computation is independent -- of the implementation and shall be: _ _ (1) Document contribution: Multiply the _ _ size of each document in octets by the _ _ _ _ number of document copies of that document. _ _ (2) Add each document contribution _ _ together. _ _ _ _ (3) Job copy contribution: Multiply the _ _ job size by the number of job copies. _ _ _ _ _ _ (4) Round up the result to the next higher K (1024 multiple). _ _ _ _ -- The total K octets to be processed can be -- used in the denominator with the

Bergman, Hastings, Isaacson, Lewis

[Page 48]

-- jmJ jobKOctetsCompleted attribute in the -- numerator in order to produce a -- 'thermometer' that indicates the progress of -- the job. _ _ -- The value (-2) means unknown. jobKOctetsCo -- Integer: The number of K (1024) octets mpleted(3121 -- currently processed by the server or device, -- including document and job copies. For), -- printing, the completed count only includes -- processing (interpreting) and markingif the -- implementation distinguishes between the -- processing and printing states; otherwise, -- the completed count includes both processing -- (interpreting) and marking combined -- together. For scanning, the completed count -- only includes scanning, if the -- implementation distinguishes between the -- processing and (to be registered) scanning -- states; otherwise the completed count -- includes both scanning and processing -- (formatting). _ _ -- The agent shall round the actual number of -- octets completed up to the next higher K. -- Thus 0 octets is represented as 0, 1-1023, -- is represented as 1, 1024-2047 is 2, etc. -- When the job completes, the values of the -- jobKOctetsRequestedTotal and the -- jmJobKOctetsCompleted attributes shall be -- equal. _ _ -- For multiple copies generated from a single -- data stream, the value shall be incremented -- as if each copy was printed from a new data -- stream without resetting the count between -- copies. See the pagesCompletedCurrentCopy -- attribute that is reset on each document copy. The total K octets completed can be used in the numerator with the jobKOctetsRequestedTotal attribute in the denominator in order to produce a "thermometer" that indicates the progress of the job. The value of this attribute shall be 0 if processing has not started for this job.

Bergman, Hastings, Isaacson, Lewis

[Page 49]

```
-- Impression attributes
_ _
-- For a print job, an impression is the marking of the
-- entire side of a sheet. Two-sided processing involves two
-- impressions per sheet. Two-up is the placement of two
-- logical pages on one side of a sheet and so is still a
-- single impression.
impressionss -- Integer: The number of impressions spooled
pooled(3222) -- to the server or device for the job so far.
impressionss -- Integer: The number of impressions sent to
entToDevice( -- the device for the job so far.
33<del>23</del>),
impressionsI -- Integer: The number of impressions
nterpreted(3 -- interpreted for the job so far.
424).
impressionsR -- Integer: The number of impressions
equested(352 -- requested by this job to produce.
<del>5</del>),
impressionsC -- Integer: The total number of impressions
ompleted(362 -- completed by the device for this job so far.
             -- For printing, the impressions completed
<del>6</del>),
             -- includes interpreting, marking, and stacking
             -- the output. For other types of job
             -- services, the number of impressions
             -- completed includes the number of impressions
             -- processed.
             ___
             -- The value of this attribute shall be 0 if
             -- processing has not started for this job.
impressionsC -- Integer: The number of impressions
ompletedCurr -- completed by the device for the current copy
entCopy(3727 -- of the current document so far. For
             -- printing, the impressions completed includes
),
             -- interpreting, marking, and stacking the
             -- output. For other types of job services,
             -- the number of impressions completed includes
             -- the number of impressions processed.
             -- The value of this attribute shall be 0 if
             -- processing has not started for this job.
-- Page attributes
_ _
-- A page is a logical page. Number up can impose more than
-- one page on a single side of a sheet. Two-up is the
-- placement of two logical pages on one side of a sheet so
-- that each side counts as two pages.
```

pagesRequest -- Integer: The number of logical pages ed(3828), -- requested by the job to be processed. pagesComplet -- Integer: The total number of logical pages -- completed for this job so far. ed(3929), pagesComplet -- Integer: The number of logical pages edCurrentCop -- completed for the current copy of the **y(4030),** -- document so far. This value shall beis -- reset to **0** for each document in the job and -- for each document copy. -- Sheet attributes _ _ -- The sheet is a single piece of a medium, whether printing -- on one or both sides. sheetsReques -- Integer: The total number of medium sheets ted(4131), -- requested to be processed for this job. **sheetsComple** -- Integer: The total number of medium sheets ted(4232), -- that have been completed marking and -- stacking for the entire job so far whether -- those sheets have been processed on one side -- or on both. -- The value of this attribute shall be 0 if processing has not started for this job. sheetsComple -- Integer: The number of medium sheets that tedCurrentCo -- have been completed marking and stacking for py(4333), -- the current copy of a document in the job so -- far whether those sheets have been processed -- on one side or on both. -- The value of this attribute shall be reset to **0** each document in the job and for each document copy if processing has not started for this job. mediumReques -- Octets: The name of the medium that is ted(4434), -- required by the job. _ _ -- A row with this attribute item may appear -- more than once in the **jmAttributeTable** for a -- job, but the jmAttributeValueAsOctets shall -- be different for each such row. mediumConsum -- Octets: The name of the medium AND ed(4535), ___

-- Integer: the number of sheets that have -- been consumed so far whether those sheets -- have been processed on one side or on both. -- This attribute shall have both values. -- A row with this attribute item may appear -- more than once in the **jmAttributeTable** for a -- job, but the jmAttributeValueAsOctets shall -- contain a different name for each such row. _ _ -- The value of this attribute shall be 0 if -- processing has not started for this job. **colorantRequ** -- Integer: The index (**prtMarkerColorantIndex**) estedIndex(4 -- in the Printer MIB of the colorant 636), -- requested. _ _ -- A row with this attribute item may appear -- more than once in the **jmAttributeTable** for a -- job, but the jmAttributeValueAsOctets shall -- be different for each such row. **colorantRequ** -- Octets: The name of the colorant requested. estedName(47 ___ 37), -- A row with this attribute item may appear -- more than once in the **jmAttributeTable** for a -- job, but the jmAttributeValueAsOctets shall -- be different for each such row. _ _ **colorantCons** -- Integer: The index (**prtMarkerColorantIndex**) umedIndex(48 -- in the Printer MIB of the colorant consumed. 38), _ _ -- A row with this attribute item may appear -- more than once in the **jmAttributeTable** for a -- job, but the jmAttributeValueAsOctets shall -- be different for each such row. colorantCons -- Octets: The name of the colorant consumed. umedName(493 ---- A row with this attribute item may appear 9), -- more than once in the **jmAttributeTable** for a -- job, but the **jmAttributeValueAsOctets** shall -- be different for each such row. _ _ -- Time attributes ___ -- tTwo forms of time are provided. Each form is represented -- in a separate attribute. Implementations may choose the -- more appropriate form. An implementation need not provide -- both forms and is recommended not to provide both forms -- for a particular attribute. However, some attributes may

```
-- be in one form and others may be in the other form,
-- depending on the source of the time. The two forms are:
-- DateAndTime is an 8 or 11 octet binary encoded year,
-- month, day, hour, minute, second, deci-second with
-- optional offset from UTC. See SNMPv2-TC.
_ _
-- NOTE: DateAndTime is not printable characters; it is
-- binary.
-- JmTimeTCStamp is the time of day measured in the number of
-- seconds as an offset from the integer value of sysUpTime
-- (which is measured in hundredths of a second).
-- See page 32.
jobSubmissio -- Octets: The date and time that the job was
nDateAndTime -- submitted. The value shall be specified
            -- using the DateAndTime textual convention
(5040),
             -- from SMIv2-TC.NOTE: DateAndTime is not
                printable characters.
jobSubmissio -- Integer: The time that the job was
nTimeStamp(5 -- submitted. The value shall be specified
           -- using the JmTimeTC textual convention from
141),
             -- SMIv2 TC (see page 32).
jobStartedBe -- Integer: The time that the job started
ingHeldTime( -- being held, i.e., the time that the job
             -- entered the held state most recently. The
52),
             -- value shall be specified using the JmTimeTC
             -- textual convention (see page 32). If the
             -- job has never entered the held state, then
             -- the value shall be 0.
jobStartedPr -- Octets: The date and time that the job
ocessingDate -- started processing. The value shall be
AndTime(5342 -- specified using the DateAndTime textual
            -- convention from SMIv2-TC.
),
jobStartedPr -- Integer: The time that the job started
ocessingTime -- processing. The value shall be specified
Stamp(5443), -- using the JmTimeTCStamp textual convention
             -- from SMIv2 TC (see page 32).
jobCompleted -- Octets: The date and time that the job
DateAndTime( -- completed processing and the medium is
5544),
             -- completely stacked in the output bin, i.e.,
             -- when the job entered the completed state.
             -- The value shall be specified using the
             -- DateAndTime textual convention from SMIv2-
             -- TC.
```

jobCompleted Time Stamp (56 45),	 Integer: The time that the job completed processing and the medium is completely stacked in the output bin, i.e., when the job entered the completed state. The value shall be specified using the JmTimeTC Stamp textual convention from SMIv2 TC (see page 32).
processingCP UTime(<mark>5746)</mark>	Integer: The amount of CPU time that the job has been processing in seconds. If the job needs attention, that elapsed time shall not be included. In other words, the processingCPUTime should be relatively repeatable.
	The value of this attribute shall be 0 if processing has not started for this job.
}	
STATUS cu DESCRIPTION "Specifies submitted represente type so th such as se with only service mi case, thre correspond Whether th job submis server or either imp for this of One of the filter out printer op printing. category. The follow assigned a jobService	rrent the type(s) of service to which the job has been (print, fax, scan, etc.). The service type is d as an enum that is bit encoded with each job service at more general and arbitrary services can be created, rvices with more than one destination type, or ones a source or only a destination. For example, a job ght scan, faxOut, and print a single job. In this e bits would be set in the jobServiceTypes attribute, ing to the values: $8+32+4=44$, respectively. is object is set from a job attribute supplied by the sion client or is set by the recipient job submission device depends on the job submission protocol. With lementation, the agent shall return a non-zero value bject indicating the type of the job. purposes of this object is to permit a requester to jobs that are not of interest. For example, a erator may only be interested in jobs that include That is why the object is in the job identification ing service component types are defined and are separate bit value in the enum for use with the Types attribute:"
This is a t	ype 2 enumeration. See Section 6.1 on page 28.

 $\begin{array}{c} 1040 \\ 1041 \\ 1042 \\ 1043 \\ 1044 \\ 1045 \\ 1046 \\ 1047 \\ 1048 \\ 1049 \\ 1050 \end{array}$

1055	SYN	TAX INTEGER other(1),	{ 	The job contains some document production instructions that are not one of the identified types.
		unknown(2),	 	The job contains some document production instructions whose type is unknown to the agent.
		<pre>print(4),</pre>	 	The job contains some document production instructions that specify printing
		<pre>scan(8),</pre>	 	The job contains some document production instructions that specify scanning
		<pre>faxIn(16),</pre>	 	The job contains some document production instructions that specify receive fax
		<pre>faxOut(32),</pre>	 	The job contains some document production instructions that specify sending fax
		getFile(64),	 	The job contains some document production instructions that specify accessing files or documents
		<pre>putFile(128),</pre>	 	The job contains some document production instructions that specify storing files or documents
		mailList(256)	 	The job contains some document production instructions that specify distribution of documents using an electronic mail system.
1056 1057	}			

1058

1059	
1060	
1061	
1062	STATUS current
1062	
1067	"This textual-convention is used in the ichStatePeacons
1065	attribute to provides additional information regarding the
1065	in TobCurrent State object and the jobState attribute. The
1067	iobstateReagons attribute identifies the reason or reasons that
1068	the job is in the propresenting held pending processing
1060	printing needs ttention canceled payed
1007	interrupted terminating retained or completed state. The
1070	server shall indicate the particular reason(s) by setting the
1071	value of the jobStatePeacong attribute. While the job states
1072	cannot be added to without impacting deployed clients it is the
1073	intent that additional .Tm.TobStatePeagongTC enums can be defined
1075	without impacting deployed clients. In other words, the
1076	.Tm.TobStateReasonsTC is intended to be extensible
1077	Subobbeatereabonble is intended to be extensible.
1078	When the job does not have any reasons for being in its current
1079	state the server shall set the value of the jobStateReasons
1080	attribute to a bit string containing all zeros
1081	accribace to a bit bering concarning arr herop.
1082	Bits in the bit string are assigned starting with the most
1083	significant bit in the most significant octet which is called
1084	bit 1 . Bit 2 is the next most significant bit in the most
1085	significant octet, etc. Bit 9 is the most significant bit in
1086	the second most significant octet, etc., up to the maximum bit:
1087	$504 (= 8 \times 63)$.
1088	
1089	An agent need only return the most significant octet up to the
1090	least significant octet that contains a non-zero bit.
1091	
1092	If all bits are zero, the agent may return an OCTET STRING of
1093	zero length. Alternatively, an agent may always return a fixed
1094	number of octets starting with the most significant octet and
1095	running through the least significant octet that could ever have
1096	a one bit in it for that implementation.
1097	
1098	This object is a type 2 bit string. See Section 6 entitled
1099	'IANA Considerations' on page 28.
1100	
1101	The following standard values are defined as bit numbers, not
1102	enums (the bit number equals the last arc of DPA id-val-reasons-
1103	xxx OID for the reasons that are in ISO DPA):"
1104	
1105	This is a type 2 bit string. See section 6.2 on page 29.
1106	SYNTAX INTEGER {
1107	really OCTET STRING(SIZE(063))
	documentsNeeded(1), The job is in the held state because
	the server or printer is waiting for
	the job's files to start and/or finish

	being transferred before the job can be scheduled to be printed.
jobHoldSet(2),	 The job is in the held state because the client specified that the job is to be held.
jobProcessAfterSpeci fied(3),	 The job is in the held state because the client specified a time specification reflected in the value of the job's jobProcessAfterDateAndTime attribute that has not yet occurred.
requiredResourcesNot Ready(4),	 The job is in the held state because 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.
<pre>successfulCompletion (5),</pre>	 The job is in the retained or completed state having completed successfully.
<pre>completedWithWarning s(6),</pre>	 The job is in the canceledterminating, retained, or completed states having completed with warnings.
<pre>completedWithErrors(7),</pre>	 The job is in the canceledterminating, retained, or completed states having completed with errors (and possibly warnings too).
<pre>cancelledByUser(8),</pre>	 The job is in the canceledterminating, retained, or completed states having been canceled by the user.
<pre>cancelledByOperator(9),</pre>	 The job is in the canceledterminating, retained, or completed states having been cancelled by the operator.
<pre>abortedBySystem(10),</pre>	 The job is in the canceledterminating, retained, or completed states having been aborted by the system.
<pre>logfilePending(11),</pre>	The job's logfile is pending file transfer.
<pre>logfileTransferring(12),</pre>	The job is in the canceledterminating, retained, or completed states and the job's logfile is being transferred

cascaded(13),	After the outbound gateway retrieves all job and document attributes and data, it stores the information into a spool directory. Once it has done this, it sends the supervisor a job processing event with this job-state reason which tells the supervisor to transition to a new job state.
<pre>deletedByAdministrat or(14),</pre>	 The administrator has issued a Delete operation on the job or a Clean operation on the server or queue containing the job; therefore the job may have been cancelled before or during processing, and will have no retention-period or completion-period.
<pre>discardTimeArrived(1 5),</pre>	<pre> The job has been deleted (cancelled with the job-retention-period set to 0) due to the fact that the time specified by the job's job-discard- time has arrived [if the job had already completed, the only action that would have occurred is that the job-retention-period would be set to 0 and the job is deleted].</pre>
<pre>postProcessingFailed (16),</pre>	 The post-processing agent failed while trying to log accounting attributes for the job; therefore the job has been placed into completedretained state with the retained jobStateReasons attribute value for a system-defined period of time, so the administrator can examine it, resubmit it, etc. The post-processing agent is a plug-and-play mechanism which the system and the customer uses to add functionality that is executed after a job has finished processing.
<pre>submissionInterrupte d(17),</pre>	Indicates that the job was not completely submitted for the following reasons: (1) the server has crashed before the job was closed by the client. The server shall put the job into the completed state (and shall not print the job). (2) the server or the document transfer method has crashed in some non-recoverable way before the document data was entirely transferred to the server. The server shall put the job into the completed

	<pre> state (and shall not print the job). (3) the client crashed or failed to close the job before the time-out period. The server shall close the job and put the job into the held state with job-state-reasons of submission-interrupted and job-hold- set and with the job's job-hold attribute set to TRUE. The user may release the job for scheduling by issuing a job submission or management protocol operation.</pre>
<pre>maxJobFaultCountExce eded(18),</pre>	The job has been faulted and returned by the server several times and that the job-fault-count exceeded the device's (or server's, if not defined for the device) cfg-max-job-fault- count. The job is automatically put into the held state regardless of the hold-jobs-interrupted-by-device- failure attribute. This job-state- reasons value is used in conjunction with the job-interrupted-by-device- failure value.
devicesNeedAttention TimeOut(19),	 One or more document transforms that the job is using needs human intervention in order for the job to make progress, but the human intervention did not occur within the site-settable time-out value and the server/device has transitioned the job to the held state.
needsKeyOperatorTime Out(20),	One or more devices or document transforms that the job is using need a specially trained operator (who may need a key to unlock the device and gain access) in order for the job to make progress, but the key operator intervention did not occur within the site-settable time-out value and the server/device has transitioned the job to the held state.
jobStartWaitTimeOut(21),	The server/device has stopped the job at the beginning of processing to await human action, such as installing a special cartridge or special non- standard media, but the job was not resumed within the site-settable time- out value and the server/device has

T

I

	 transitioned the job to the held state. Normally, the job is resumed by means outside the job submission protocol, such as some local function on the device.
jobEndWaitTimeOut(22)),	The server/device has stopped the job at the end of processing/printing to await human action, such as removing a special cartridge or restoring standard media, but the job was not resumed within the site-settable time- out value and the server/device has transitioned the job to the completedretained state. Normally, the job is resumed by means outside the job submission protocol, such as some local function on the device, whereupon the job shall transition immediately to the canceledterminating state.
jobPasswordWaitTimeO ut(23),	The server/device has stopped the job at the beginning of processing to await input of the job's password, but the human intervention did not occur within the site-settable time-out value and the server/device has transitioned the job to the held state. Normally, the password is input and the job is resumed by means outside the job submission protocol, such as some local function on the device.
<pre>deviceTimedOut(24),</pre>	A device that the job was using has not responded in a period specified by the device's site-settable attribute.
connectingToDeviceTi meOut(25),	 The server is attempting to connect to one or more devices which may be dial- up, polled, or queued, and so may be busy with traffic from other systems, but server was unable to connect to the device within the site-settable time-out value and the server has transitioned the job to the held state.
<pre>transferring(26),</pre>	T he job is being transferred to a down stream server or device.
<pre>queuedInDevice(27),</pre>	The job has been queued in a down
тт, т т т	$[\mathbf{D}_{1}, \mathbf{c}, \mathbf{c}]$

-- stream server or device. jobCleanup(28), -- The server/device is performing -- cleanup activity as part of ending -- normal processing. processingToStopPoin -- The requester has issued an operation -- to interrupt the job and the t(29), -- server/device is processing up until -- the specified stop point occurs. jobPasswordWait(30), -- The server/device has selected the job -- to be next to process, but instead of -- assigning resources and started the -- job processing, the server/device has -- transitioned the job to the **held** state -- to await entry of a password (and -- dispatched another job, if there is -- one). The user resumes the job either -- locally or by issuing a remote -- operation and supplying a job--- password=secret-code input parameter -- that must match the job's job-password -- attribute. validating(31), -- The server/device is validating the -- job after accepting the job. The job -- state may be creating, held, pending, -- or processing. -- The operator has held the entire queue queueHeld(32), -- by means outside the scope of the Job -- model. jobProofWait(33), -- The job has produced a single proof -- copy and is in the **held** state waiting -- for the requester to issue an -- operation to release the job to print -- normally, obeying the **job-copies** and -- copy-count job and document attributes -- that were originally submitted. **heldForDiagnostics(3** -- The system is running intrusive 4), -- diagnostics, so the all jobs are being -- held. serviceOffLine(35), -- The service/document transform is off--- line and accepting no jobs. All -- pending jobs are put into the held -- state. This could be true if its -- input is impaired or broken. noSpaceOnServer(36), -- The job is held because there is no Bergman, Hastings, Isaacson, Lewis [Page 61]

	 room on the server to store all of the job. For example, there is no room for the document data or a scan-to- file job.
pinRequired(37),	 The System Administrator settable device policy is (1) to require PINs, and (2) to hold jobs that do not have a pin supplied as an input parameter when the job was created. The requester shall either (1) enter a pin locally at the device or issue a remote operation supplying the PIN in order for the job to be able to proceed.
exceededAccountLimit (38),	The account for which this job is drawn has exceeded its limit. This condition should be detected before the job is scheduled so that the user does not wait until his/her job is scheduled only to find that the account is overdrawn. This condition may also occur while the job is processing either as processing begins or part way through processing.
	 An overdraft mechanism should be included to be user-friendly, so as to minimize the chances that the job cannot finish or that media is wasted. For example, the server/device should finish the current copy for a job with collated document copies, rather than stopping in the middle of the current document copy.
heldForRetry(39),	The job encountered some errors that the server/device could not recover from with its normal retry procedures, but the error is worth trying the job later, such as phone number busy or remote file system in-accessible. For such a situation, the server/device shall add the held-for-retry value to the job's jmJobStateReasons attributeobject and transition the job from the processing to the held, rather than to the completedretained state.

-- The following values are from the X/Open PSIS draft standard:

cancel l edByShutdown(40),	The job was cancelled because the server or device was shutdown before completing the job. The job shall be placed in the pending state [if the job was not started, else the job shall be placed in the terminating state].
deviceUnavailable(41),	 This job was aborted by the system because the device is currently unable to accept jobs. This reason [shall be] used in conjunction with the reason aborted-by-system. The job shall be placed in the pending state.
wrongDevice(42),	 This job was aborted by the system because the device is unable to handle this particular job; the spooler should try another device. This reason [shall be] used in conjunction with the reason aborted-by- system. The job shall be pending if the queue contains other physical devices that the job could print on, and the spooler is capable of not sending the job back to a physical device that has rejected the job for this job-state- reasons value. Otherwise, [the job] shall be placed in the completed state with the jmJobStateReasons retained value set in the jobStateReasons attribute.
badJob(43),	 This job was aborted by the system because this job has a major problem, such as an ill-formed PDL; the spooler should not even try another device. This reason shall be used in conjunction with the reason aborted- by-system. The job shall be placed in the terminating state.
jobInterruptedByDevi ceFailure(44),	 A device or the print system software that the job was using has failed while the job was processing. The device is keeping the job in the held state until an operator can determine what to do with the job.

-- The following additional job state reasons have been added to align -- specify sub-states of the held state that are in ISO DPA:with the

Internet Printing Protocol (IPP):

jobPrinting(45)	 The job is putting marks on a medium. This optional job state reason is provided for systems where there is a significant difference in the time period while a job is in the processing state between putting marks on a medium and other activities, such as interpreting the document data. For systems that interpret and mark at the same time for a job need not implement this job state reason.
jobPreProcessing(45) ,	 The job has been created on the server or device but the submitting client is in the process of adding additional job components and no documents have started processing. The job maybe in the process of being checked by the server/device for attributes, defaults being applied, a device being selected, etc.
jobPaused(46),	 The job has been indefinitely suspended by a client issuing an operation to suspend the job so that other jobs may proceed using the same devices. The client may issue an operation to resume the paused job at any time, in which case the server or printer places the job in the held or pending states and the job is eventually resumed at the point where the job was paused.
jobInterrupted(47),	 The job has been interrupted while processing by a client issuing an operation that specifies another job to be run instead of the current job. The server or printer will automatically resume the interrupted job when the interrupting job completes.
jobRetained(48)	<pre> The job is being retained by the server or printer after processing and all of the media have been successfully stacked in the output bin(s). The job (1) has completed successfully or with warnings or errors, (2) has</pre>

Bergman, Hastings, Isaacson, Lewis

[Page 64]

```
-- been aborted while printing by the
-- server/device, or (3) has been
-- cancel<del>l</del>ed by the submitting user or
-- operator before or during processing.
-- The job's jmJobStateReasons
-- attributeobject shall contain the
-- reasons that the job has entered the
-- retained sub-state of the completed
-- state.
___
-- While in the retained state, all of
-- the job's document data (and submitted
-- resources, such as fonts, logos, and
-- forms, if any) are retained by the
-- server or device; thus a client could
-- issue an operation to resubmit the job
-- (or a copy of the job) while the job
-- is in the retained state.
___
-- The retained state is conditionally
-- mandatory. Implementations that do
-- not retain jobs after they are
-- finished processing such that the
   client could request that the job be
   repeated (or resubmitted), need not
   implement the retained state.
```

1108

}

1109 1110

-- The following table shows the **JmJobStateReasonsTC** values and the -- job states for which they are applicable. The ISO DPA job state -- reasons are shown along with additional job-state-reasons that -- give users additional feedback on the progress of their job:

1111

1112 Table 1 - Legal Job States for each Job State Reason

 Descriptive Name	Allowed job states
 documents-needed(1)	held
 job-hold-set(2)	held
 job-process-after-specified(3)	held
 required-resources-not-ready(4)	held
 successful-completion(5)	completed
 completed-with-warnings(6)	completed
 completed-with-errors(7)	completed
 cancel l ed-by-user(8)	canceled
 cancel l ed-by-operator(9)	canceled
 aborted-by-system(10)	canceled
 logfile-pending(11)	canceled
 logfile-transferring(12)	canceled
 cascaded(13)	canceled
 deleted-by-administrator(14)	canceled
 discard-time-arrived(15)	canceled
 <pre>postprint-failed(16)</pre>	canceled, completed
 submission-interrupted(17)	canceled
 <pre>max-job-fault-count-exceeded(18)</pre>	canceled
 devices-need-attention-time-out(19)	held, canceled
 needs-key-operator-time-out(20)	held, canceled
 job-start-wait-time-out(21)	canceled
 job-end-wait-time-out(22)	canceled
 job-password-wait-time-out(23)	held, pending
 device-timed-out(24)	held, canceled
 connecting-to-device-time-out(25)	held, canceled
 transferring(26)	processing
 queued-in-device(27)	processing
 job-cleanup(28)	processing
 processing-to-stop-point(29)	processing
 job-password-wait(30)	held, processing
 validating(31)	held, pending, processing
 queue-held(32)	held
 job-proof-wait(33)	held
 held-for-diagnostics(34)	held
 <pre>service-off-line(35)</pre>	held
 no-space-on-server(36)	held
 pin-required(37)	held, canceled
 exceeded-account-limit(38)	held, canceled

Job Monitoring MIB, V0.<u>8</u>71

 Descriptive Name	Allowed job states
 held-for-retry(39)	held
 canceledByShutdown(40)	canceled
 deviceUnavailable(41)	pending
 wrongDevice(42)	canceled
 badJob(43)	canceled
 jobInterruptedByDeviceFailure(44)	held
 job-printing(45)	processing
jobPreProcessing(45)	held
 jobPaused(46)	held
 jobInterrupted(47)	held
 jobRetained(48)	completed

1113 1114

```
1115
1116
      -- The General Group (Mandatory)
1117
1118
      -- The jmGeneralGroup consists entirely of the jmGeneralTable.
1119
1120
      -- Implementation of every object in this group is mandatory.
1121
      -- See Section 4 entitled 'Conformance Considerations' on page 26.
1122
1123
      jmGeneral OBJECT IDENTIFIER ::= { jobmonmib 5 }
1124
1125
      jmGeneralTable OBJECT-TYPE
1126
                      SEQUENCE OF JmGeneralEntry
          SYNTAX
1127
         MAX-ACCESS not-accessible
1128
          STATUS
                      current
1129
          DESCRIPTION
1130
              "The jmGeneralTable consists of information of a general nature
1131
              that are per-job-set, but are not per-job. See Terminology and
1132
              Job Model on page 10 for the definition of a job set.
1133
1134
              The jmGeneralTable which is indexed by:
1135
1136
              1.
                  jmJobSetIndex - a running index of Job Set instances
1137
                  supported by this device or server. A job set is used in
                  the MIB to represent the separation of jobs into disjoint
1138
1139
                  sets for scheduling purposes in a server, typically into
1140
                  separate job queues. See Terminology and Job Model on page
1141
                  10 for the definition of a job set."
1142
          ::= \{ jmGeneral 1 \}
1143
1144
      jmGeneralEntry OBJECT-TYPE
1145
          SYNTAX
                      JmGeneralEntry
1146
         MAX-ACCESS not-accessible
1147
          STATUS
                      current
1148
          DESCRIPTION
1149
              "Information about a job set (queue). See Terminology and Job
1150
              Model on page 10 for the definition of a job set.
1151
1152
              An entry shall exist in this table for each job set."
1153
          INDEX { jmJobSetIndex }
1154
          ::= { jmGeneralTable 1 }
1155
1156
     JmGeneralEntry ::= SEQUENCE {
          jmJobSetIndex
                                              Integer32(1...32767),
          jmGeneralJobSetName
                                              OCTET STRING(SIZE(0..63))
          jmGeneralJobPersistenceCompletedP
                                              Integer32(0..2147483647),
          olicy
          jmGeneralAttributePersistence
                                              Integer32(0..2147483647),
          jmGeneralMaxNumberOfJobs
                                              Integer32(0..2147483647),
          jmGeneralNumberOfActiveJobsToComp
                                              Integer32(0..2147483647),
          lete
          imGeneralOldestActiveJobIndex
                                              Integer32(0..2147483647),
          jmGeneralNewestActiveJobIndex
                                              Integer32(0..2147483647),
          jmGeneralNumberOfJobsCompleted
                                              Integer32(0..2147483647)
```

[Page 68]

Job Monitoring MIB, V0.<mark>871</mark>

1157 } 1158 1159 jmJobSetIndex OBJECT TYPE 1160 SYNTAX Integer32(1...32767) 1161 MAX ACCESS not accessible 1162 STATUS current 1163 DESCRIPTION "The 16 bit index of a Job Set instance used to represent the 1164 separation of jobs into disjoint sets for scheduling purposes in 1165 1166 a server, typically into separate job queues. See for the definition of a job set. Agents implementing a single Job Set 1167 1168 instance shall use an index value of 1 for this object." 1169 \div = { \neg mGeneralEntry 1 } 1170 1171 jmGeneralJobSetName OBJECT-TYPE 1172 SYNTAX OCTET STRING(SIZE(0..63)) 1173 MAX-ACCESS read-only 1174 STATUS current 1175 DESCRIPTION 1176 "The human readable administratively assigned name of this job 1177 set. Typically, this name will be the name of the job queue. 1178 If a server or printer has only a single job set, this object can be the administratively assigned name of the server or 1179 printer itself. This name does not need to be unique, though 1180 1181 each job set in a single Job Monitoring MIB should have distinct 1182 names. 1183 1184 The purpose of this object is to help the user of the job 1185 monitoring application distinguish between several job sets in 1186 implementations that support more than one job set." 1187 ::= { jmGeneralEntry 12 } 1188 1189 jmGeneralJobPersistenceCompletedPolicy OBJECT-TYPE 1190 Integer32(0..2147483647) SYNTAX 1191 MAX-ACCESS read-only 1192 STATUS current 1193 DESCRIPTION 1194 "The minimum time in seconds that an entry will remain the 1195 device or server keeps jobs in the jmJobIDTable and 1196 jmJobStateCompletedTable after processing/printing has completed 1197 as specified by the system administrator or the implementation 1198 for this instance of the Job Set." 1199 ::= { jmGeneralEntry 23 } 1200 1201 jmGeneralAttributePersistence OBJECT-TYPE 1202 SYNTAX Integer32(0..2147483647) 1203 read-only MAX-ACCESS 1204 STATUS current 1205 DESCRIPTION 1206 "The minimum time in seconds that an entry will remain in the jmAttributeTable after processing/printing has completed, i.e., 1207 1208 the time in seconds starting when the job enters the completed 1209 or **canceled** state. The value of this object may be either (1)

Job Monitoring MIB, V0.<u>8</u>71

1210	set by the system administrator by means outside this
1211	specification or may be (2) fixed by the
1212	implementationimplementtation for this instance of the Job Set,
1213	depending on implementation. This value shall be equal to or
1214	less than the value of jmGeneralJobPersistence. Attributes that
1215	are shared between the jmJobIDTable/jmJobStateTable and the
1216	jmAttributeTable shall be governed by the larger value in all
1217	tables."
1218	::= { jmGeneralEntry 34 }
1219	
1220	jmGeneralMaxNumberOfJobs OBJECT TYPE
1221	SYNTAX Integer32(02147483647)
1222	MAX ACCESS read only
1223	STATUS current
1224	DESCRIPTION
1225	"The maximum number of queued and completed jobs that this
1226	server or print can support at the same time.
1227	
1228	The value (-1) indicating other shall indicate that there is no
1229	fixed limit."
1230	∶∶= { jmGeneralEntry 4 }
1231	
1232	jmGeneralNumberOfActiveJobsToComplete OBJECT-TYPE
1233	SYNTAX Integer32(02147483647)
1234	MAX-ACCESS read-only
1235	STATUS current
1236	DESCRIPTION
1237	"The <u>currenttotal number of <u>active</u> jobs currently in the</u>
1238	jmJobIDTable, jmJobStateTable, and jmAttributeTable that are to
1239	be completed, i.e., the total number of jobs that have neither
1240	completed nor have been canceled are in the following states:
1241	pre-processing, held, pending, processing, needs-attention,
1242	paused, interrupted, or terminating, but not retained or
1243	completed. See JmJobStateTC on page 32 for the exact
1244	specification of the semantics of the job states.
1245	
1246	It there are no active jobs, the value of this object shall be
1247	
1248	::= { jmGeneralEntry <u>4</u> 5 }
1249	
1250	JINGENERALNUMBERUIJOBSCOMPLETED OBJECT TYPE
1251	SYNIAX Integer32(U214/48364/)
1252	MAA ACCESS ICau Only
1233	DECONTRACTOR
1254	UEDCKIFIION
1255	appropriated is the total number of jobs that are
1250	following states: mateined as sempleted, but not me
1257	hold ponding prograging product the pre-processing,
1250	meru, penarny, processiny, meeds-attention, paused, interrupted,
1227	or cerminating. See for the exact specification of the
1200	semantics of retained, completed and the other states.
1201	

Job Monitoring MIB, V0.<u>8</u>71

1262	The value of the jmGeneralNumberOfJobsCompleted shall equal the
1263	number of jobs in the jmCompletedTable . The sum of
1264	jmGeneralNumberOfJobsToComplete and
1265	jmGeneralNumberOfJobsCompleted shall be equal to the number of
1266	jobs in the jmJobTable."
1267	÷÷= { jmGeneralEntry 6 }
1268	
1269	jmGeneralOldestActiveJobIndex OBJECT-TYPE
1270	SYNTAX Integer32 (02147483647)
1271	MAX-ACCESS read-only
1272	STATUS current
1273	DESCRIPTION
1274	"The jmJobIndex of the oldest active job, i.e., the job in the
1275	jmJobStateTable and jmAttributeTable that has been there the
1276	longest and has neither completed nor been canceled .
1277	
1278	If there are no active jobs, the value shall be 0.
1279	
1280	NOTE - For implementations that process jobs in order of
1281	submission, this object indicates the 'separating line' between
1282	completed jobs and jobs that are still active. However, an
1283	application shall still have to skip over canceled jobs when
1284	searching for active jobs.
1285	
1286	NOTE - Applications that wish to skip over completed or canceled
1287	jobs may use this value to start with the oldest active job and
1288	continue until they reach the index value equal to
1289	jmGeneralNewestActiveJobIndex, skipping over any completed or
1290	canceled jobs that might intervene. Since jobs may arrive while
1291	such an application is performing GetNext operations, the
1292	application should always get the value of
1293	jmGeneralNewestActiveJobIndex in each GetNext operation to see
1294	if this job is still the newest. If an application gets the no
1295	more rows ??? return, the job index may have wrapped such that
1296	the JmGeneralNewestActiveJobIndex is smaller than
1297	JmGeneralOldestActiveJobIndex. In this case, the application
1298	shall start over at 1 and continue the GetNext operations to
1299	<u>ind the rest of the active jobs."</u>
1300	::= { jmgeneralEntry 50 }
1301	
1202	JINGENERALNEWESTACTIVEJODINDEX OBJECT-TYPE
1202	MAX ACCERCE mood only
1304	MAX-ACCESS read-only
1305	
1300	UEBCRIPTION "The im TebInder of the new ort active job is the job in the
1307	intopstateTable and inAttributeTable that has been added most
1300	recently and has neither completed nor been canceled
1310	Tecentry and map nerther compreted nor been cancered.
1310	If there are no active jobs the value shall be 0 "
1317	$::= \{ imGeneralEntry 6 \}$
1312	··- []mgenerarmitry 0]
1010	

```
1314
      - The Queue Group (Conditionally Mandatory)
      - The jmQueueGroup consists of job objects that are needed by a
      - server or device that queues jobs, but are not needed after the
      - job has completed processing, i.e., are not needed by accounting
      - applications.
      - The jmQueueGroup is conditionally mandatory meaning that the
     - jmQueueGroup shall be implemented by a Job Monitoring MIB agent
      - that is instrumenting a server or printer that performs queuing
      - (or spooling).
      - The jmQueueGroup is made up entirely of the jmQueueTable which is
      - an ordered list of jobs in a job set that have not completed
     - processing. The jmQueueTable is indexed by:
      - 1) jmJobSetIndex - a running index of Job Set instances supported
           by this device or server. A job set is used in the MIB to
           represent the separation of jobs into disjoint sets for
           scheduling purposes in a server, typically into separate job
           queues. See on page for the definition of a job set.
      - 2) jmQueueIndex a running index of the jobs that have not
           finished processing and shall indicate the order that the jobs
      _____
           are currently scheduled to be processed.
      - Implementation of this group is conditionally mandatory, i.e.,
      - mandatory if the server or printer that the agent is instrumenting
      - queues jobs (rather than just passing the jobs through). See
      - Section entitled '' on page .
1315
1316
     jmQueue OBJECT IDENTIFIER ::= { jobmonmib 6 }
1317
1318
     jmQueueTable OBJECT TYPE
1319
         SYNTAX SEQUENCE OF JmQueueEntry
1320
         MAX ACCESS not accessible
1321
         STATUS current
1322
         DESCRIPTION
1323
             "A table of per job information needed by a server or device
1324
             that performs queuing."
1325
         \div = { jmQueue 1 }
1326
1327
     jmQueueEntry OBJECT TYPE
SYNTAX JmQueueEntry
1328
1329
         MAX ACCESS not accessible
1330
         STATUS current
1331
         DESCRIPTION
1332
             "Information about a job in a server or printer that performs
1333
             <del>queuing.</del>
1334
```
1335	An entry shall exist in this table for each job in a job set
1336	that is queued, i.e., for each job that has not completed
1337	processing."
1338	INDEX { jmJobSetIndex, jmQueueIndex }
1339	÷ += { jmOueueTable 1 }
1340	
1341	JmOueueEntry := SEOUENCE {
1011	$\frac{1}{100} = \frac{1}{100} = \frac{1}$
	$\frac{\text{Integer 52(1211,105017)}}{\text{Integer 22(1211,105017)}}$
	$\frac{1}{100000} = 1000000000000000000000000000000000000$
	$\frac{1}{100} \frac{1}{100} \frac{1}$
	im Tob Drogogg A ft or Date And Time Date And Time
1242	JIIDODFIOCESSALCEIDACEANGIIIIE DACCANGIIIIE
1342	
1343	
1344	JmQueueIndex OBJECT TYPE
1345	SYNTAX Integer32(0214/483647)
1346	MAX ACCESS not accessible
1347	STATUS current
1348	DESCRIPTION
1349	"The 32 bit index of the jobs that have not finished processing.
1350	The index values shall be assigned monatonically increasing as
1351	the server or printer determines the order of processing. The
1352	agent shall change the value of this object dynamically as the
1353	priority ordering of jobs changes. Thus the imOueueTable orders
1354	the jobs into their current priority order which can change as
1355	new jobs are submitted and/or the configuration of the Printer
1356	is changed "
1357	$\frac{15}{2} = \frac{1}{2} \frac{1}{2}$
1358	
1350	
1360	$\frac{\text{SVMTAV}}{\text{Trtogor}^2(1-21/7/826/7)}$
1360	MAX ACCECC not accordible
1262	
1262	DECONTRACTOR
1303	DESCRIPTION
1364	"The job's identifier generated by the server or device when
1365	that server or device accepted the job. This value permits the
1366	management application to access the other tables to obtain the
1367	job specific objects. This value shall be the same for a job in
1368	the jmQueueTable as the corresponding jmJobIndex value in the
1369	jmJobTable for this job.
1370	
1371	The value 0 shall not be generated. Agents instrumenting
1372	systems that contain jobs with a job identifier of 0 shall map
1373	the value 0 to a value that is one higher than the highest job
1374	identifier value that any job can have on that system."
1375	\div = { $imOueueEntry 2$ }
1376	
1377	imOueueNumberOfInterveningTobs OBJECT TYPE
1378	$\frac{\text{SVNTAX}}{\text{Thteger 32(02147483647)}}$
1379	MAX ACCESS road only
1380	CTATIC Current
1200	
1301	DEDCRIFIION

1382	"The number of jobs that are expected to be processed before
1383	this job is processed according to the implementation's queuing
1384	algorithm if no other jobs were to be submitted. The agent
1385	shall return a value of 0 for this object when the job starts
1386	processing "
1387	$\frac{1}{10000000000000000000000000000000000$
1388	
1380	in TobDriority ODIECT-TYDE
1200	$\frac{\text{OVED}}{\text{OVED}} = \frac{100 \text{ MeV}}{100 \text{ MeV}}$
1201	MAX ACCERC mood only
1391	MAX ACCESS read only
1392	STATUS current
1393	DESCRIPTION
1394	"This attribute specifies a priority for scheduling the job. It
1395	is used by servers and devices that employ a priority based
1396	scheduling algorithm.
1397	
1398	A higher value specifies a higher priority. The value 1 is
1399	defined to indicate the lowest possible priority (a job which a
1400	priority based scheduling algorithm shall pass over in favor of
1401	higher priority jobs). The value 100 is defined to indicate the
1402	highest possible priority. Priority is expected to be evenly or
1403	'normally' distributed across this range. The mapping of vendor
1404	defined priority over this range is implementation specific.
1405	
1406	A value of 0 shall be returned by implementations that do not
1/07	have a priority based queuing algorithm "
1407	··- / imOnonoEntry /]
1407 1408 1409	++= { jmQueueEntry 4 }
1407 1408 1409 1410	im Joh Progonal ft or Date and Time OR TECT TYPE
1408 1409 1410 1411	inde a profity based queating argorrena.
1407 1408 1409 1410 1411 1412	<pre>inave a priority based queuing argorithm. it = { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX_ACCESSroad_oply</pre>
1407 1408 1409 1410 1411 1412 1413	index index implosition implosition implosition i
1407 1408 1409 1410 1411 1412 1413	intervention implosition implosition implosition
1407 1408 1409 1410 1411 1412 1413 1414	<pre>indve a priority based queuing argorithm. it = { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This shippet argonifies the selender dete and time of der often "This shippet argonifies the selender dete and time of der often "This shippet argonifies the selender dete and time of der often "This shippet argonifies the selender dete and time of der often "This selender the selender dete and time of der often "This selender"</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415	interaction interaction imJobProcessAfterDateAndTime OBJECT_TYPE SYNTAX DateAndTime MAX_ACCESS read_only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416	<pre>inverse a priority based queuing argorithm. it = { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417	<pre>inve a priority based queuing argorithm. it = { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future,</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418	<pre>inve a priority based queuing argorithm. it = { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419	<pre>indve a priority based queuing argorithm.</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420	<pre>indve a priority based queuing argorithm. it = { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421	<pre>indve d priority based queuing digorithm. it= { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422	<pre>inverse priority based queuing argorithm.</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423	<pre>indve d piroffey based queuing digofferm. it = { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When the specified date and time arrives, the server shall remove the jobProcessAfterSpecified bit value from the job's</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424	<pre>inver a priority based queuing argorithm. it = { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When the specified date and time arrives, the server shall remove the jobProcessAfterSpecified bit value from the job's jmJobStateReasons object and, if no other reasons remain, shall</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424 1425	<pre>inave a priority based queuing algorithm. it= { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When the specified date and time arrives, the server shall remove the jobProcessAfterSpecified bit value from the job's jmJobStateReasons object and, if no other reasons remain, shall change the job's imJobCurrentState to pending so that the job</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424 1425 1426	<pre>inave a priority based queuing argorithm. i+= { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When the specified date and time arrives, the server shall remove the jobProcessAfterSpecified bit value from the job's jmJobStateReasons object and, if no other reasons remain, shall change the job's jmJobCurrentState to pending so that the job becomes a candidate for being scheduled on devices(s).</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424 1425 1426 1427	<pre>indve u priority based quaing argorithm. ;:= (jmQueueEntry 4) jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When the specified date and time arrives, the server shall remove the jobProcessAfterSpecified bit value from the job's jmJobStateReasons object and, if no other reasons remain, shall change the job's jmJobCurrentState to pending so that the job becomes a candidate for being scheduled on devices(s).</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424 1425 1426 1427 1428	<pre>inve u priorty based queung argorrend. ::= { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When the specified date and time arrives, the server shall remove the jobProcessAfterSpecified bit value from the job's jmJobStateReasons object and, if no other reasons remain, shall change the job's jmJobCurrentState to pending so that the job becomes a candidate for being scheduled on devices(s). The server shall assign an empty value to the</pre>
1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424 1425 1426 1427 1428 1429	<pre>indve a priority based queuing argorithm. ::= { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When the specified date and time arrives, the server shall remove the jobProcessAfterSpecified bit value from the job's jmJobStateReasons object and, if no other reasons remain, shall change the job's jmJobCurrentState to pending so that the job becomes a candidate for being scheduled on devices(s). The server shall assign an empty value to the imJobProcessAfterDateAndTime object when no process after time </pre>
$\begin{array}{c} 1407\\ 1408\\ 1409\\ 1410\\ 1411\\ 1412\\ 1413\\ 1414\\ 1415\\ 1416\\ 1417\\ 1418\\ 1419\\ 1420\\ 1421\\ 1422\\ 1423\\ 1424\\ 1425\\ 1426\\ 1427\\ 1428\\ 1429\\ 1430\\ \end{array}$	<pre>indve u priority based queung argorithm. ::= { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When the specified date and time arrives, the server shall remove the jobProcessAfterSpecified bit value from the job's jmJobStateReasons object and, if no other reasons remain, shall change the job's jmJobCurrentState to pending so that the job becomes a candidate for being scheduled on devices(s). The server shall assign an empty value to the jobProcess after pateAndTime object when no process after time has been georified on the to be been acandidate for been shall assign an empty value to the imJobProcess after pateAndTime object when no process after time has been georified at the top ish shall be acandidate for </pre>
$\begin{array}{c} 1407\\ 1408\\ 1409\\ 1410\\ 1411\\ 1412\\ 1413\\ 1414\\ 1415\\ 1416\\ 1417\\ 1418\\ 1419\\ 1420\\ 1421\\ 1422\\ 1423\\ 1424\\ 1425\\ 1426\\ 1427\\ 1428\\ 1429\\ 1430\\ 1431\\ \end{array}$	<pre>indve u priority based queuing argorithm.</pre>
$\begin{array}{c} 1407\\ 1408\\ 1409\\ 1410\\ 1411\\ 1412\\ 1413\\ 1414\\ 1415\\ 1416\\ 1417\\ 1418\\ 1419\\ 1420\\ 1421\\ 1422\\ 1423\\ 1424\\ 1425\\ 1426\\ 1427\\ 1428\\ 1429\\ 1430\\ 1431\\ 1432\\ \end{array}$	<pre>inver d priority based queuing digorithm.</pre>
$\begin{array}{c} 1407\\ 1408\\ 1409\\ 1410\\ 1411\\ 1412\\ 1413\\ 1414\\ 1415\\ 1416\\ 1417\\ 1418\\ 1419\\ 1420\\ 1421\\ 1422\\ 1423\\ 1424\\ 1425\\ 1426\\ 1427\\ 1428\\ 1429\\ 1430\\ 1431\\ 1432\\$	<pre>inverse profiley based queuing algorithm. it= { jmQueueEntry 4 } jmJobProcessAfterDateAndTime OBJECT TYPE SYNTAX DateAndTime MAX ACCESS read only STATUS current DESCRIPTION "This object specifies the calendar date and time of day after which the job shall become a candidate to be scheduled for processing. If the value of this attribute is in the future, the server shall set the value of the job's jmJobCurrentState to held and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons object and shall not schedule the job for processing until the specified date and time has passed. When the specified date and time arrives, the server shall remove the jobProcessAfterSpecified bit value from the job's jmJobStateReasons object and, if no other reasons remain, shall change the job's jmJobCurrentState to pending so that the job becomes a candidate for being scheduled on devices(s). The server shall assign an empty value to the jmJobProcessAfterDateAndTime object when no process after time has been specified, so that the job shall be a candidate for processing immediately." time { jmQueueEntry 5 } } </pre>

Job Monitoring MIB, V0.871 April 4, 1997

```
1434
      --- The Completed Group (Mandatory)
     - The jmCompletedGroup consists entirely of the jmCompletedTable
     - which is an ordered list of the jobs in the job set that have
     - completed processing, i.e., jobs that are in the terminating,
     - retained or completed state. The jmCompletedTable is indexed by:
     — 1) jmJobSetIndex - a running index of Job Set instances supported
           by this device or server. A job set is used in the MIB to
           represent the separation of jobs into disjoint sets for
           scheduling purposes in a server, typically into separate job
           queues. See for the definition of a job set.
     - 2) jmCompletedIndex a running index of the jobs that have
     _____
           finished processing.
     - Implementation of every object in this group is mandatory. See
     - Sectio
1435
1436
     jmCompleted OBJECT IDENTIFIER := { jobmonmib 7 }
1437
1438
     jmCompletedTable OBJECT TYPE
1439
         SYNTAX SEQUENCE OF JmCompletedEntry
1440
         MAX ACCESS not accessible
1441
         STATUS current
1442
         DESCRIPTION
1443
             "A table of pointers to jobs that have finished processing, have
1444
             been cancelled by a user or operator, or the system has
1445
             aborted."
1446
         \div = { jmCompleted 1 }
1447
1448
     jmCompletedEntry OBJECT TYPE
1449
         SYNTAX JmCompletedEntry
         MAX ACCESS not accessible
1450
1451
         STATUS current
1452
         DESCRIPTION
1453
             "A pointer to a job that has finished processing.
1454
1455
             An entry shall exist in this table for each job that has
1456
             finished processing, due to normal completion, cancellation by a
1457
             user, or termination by the system."
1458
         INDEX { jmJobSetIndex, jmCompletedIndex }
1459
         ++= { jmCompletedTable 1 }
1460
1461
     JmCompletedEntry ::= SEQUENCE {
     jmCompletedIndex
                                    Integer32(1..2147483647),
     jmCompletedJobIndex
                                   Integer32(1..2147483647)
1462
     +
1463
1464
     jmCompletedIndex OBJECT TYPE
1465
         SYNTAX Integer32(1..2147483647)
```

1466	MAX ACCESS not accessible
1467	STATUS current
1468	DESCRIPTION
1469	"The 32 bit index of the jobs that are in the retained or
1470	completed states. The agent shall add jobs to the end of the
1471	jmCompletedTable, so that monitor programs can quickly determine
1472	what jobs have completed since the last time that the monitoring
1473	programs accessed the jmCompletedTable. The index values shall
1474	be monatonically increasing. Therefore, the order of the jobs
1475	specified by the value of this index shall be the order in which
1476	the jobs finished processing.
1477	
1478	Since the jmCompletedIndex shall roll over when the
1479	jmCompletedIndex would have reached 2^31 (but no lower),
1480	monitoring programs shall handle such roll over."
1481	++= { jmCompletedEntry 1 }
1482	
1 100	
1483	JmCompletedJobindex OBJECT TYPE
1483 1484	JmCompletedJobindex OBJECT TYPE SYNTAX Integer32(12147483647)
1483 1484 1485	JmCompletedJobindex OBJECT TYPE SYNTAX Integer32(12147483647) MAX ACCESS not accessible
1483 1484 1485 1486	JmCompletedJobindex OBJECT TYPE SYNTAX Integer32(12147483647) MAX ACCESS not accessible STATUS current
1483 1484 1485 1486 1487	JmCompletedJobindex OBJECT TYPE SYNTAX Integer32(12147483647) MAX ACCESS not accessible STATUS current DESCRIPTION
1483 1484 1485 1486 1487 1488	JmCompletedJobIndex OBJECT TYPE SYNTAX Integer32(12147483647) MAX ACCESS not accessible STATUS current DESCRIPTION "The job's identifier generated by the server or device when
1483 1484 1485 1486 1487 1488 1488	JmCompletedJobIndex OBJECT TYPESYNTAX Integer32(12147483647)MAX ACCESS not accessibleSTATUS currentDESCRIPTION"The job's identifier generated by the server or device when that server or device accepted the job. This value permits the
1483 1484 1485 1486 1487 1488 1489 1490	JmCompletedJobIndex OBJECT TYPESYNTAX Integer32(12147483647)MAX ACCESS not accessibleSTATUS currentDESCRIPTION"The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the
1483 1484 1485 1486 1487 1488 1489 1490 1491	JmCompletedJobIndex OBJECT TYPESYNTAX Integer32(12147483647)MAX ACCESS not accessibleSTATUS currentDESCRIPTION"The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job specific objects. This value shall be the same for a job in
1483 1484 1485 1486 1487 1488 1489 1490 1491 1492	JmCompletedJobIndex OBJECT TYPESYNTAX Integer32(12147483647)MAX ACCESS not accessibleSTATUS currentDESCRIPTION"The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job specific objects. This value shall be the same for a job in the jmQueueTable as the corresponding jmJobIndex value in the
1483 1484 1485 1486 1487 1488 1487 1488 1489 1490 1491 1492 1493	JMCompletedJobIndex OBJECT TYPESYNTAXInteger32(12147483647)MAX ACCESS not accessibleSTATUScurrentDESCRIPTION"The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job specific objects. This value shall be the same for a job in the jmQueueTable as the corresponding jmJobIndex value in the jmJobTable for this job.
1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494	JMCompletedJobIndex OBJECT TYPESYNTAXInteger32(12147483647)MAX ACCESS not accessibleSTATUScurrentDESCRIPTION"The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job specific objects. This value shall be the same for a job in the jmQueueTable as the corresponding jmJobIndex value in the jmJobTable for this job.
1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495	JmCompletedJobIndex OBJECT TYPESYNTAXInteger32(12147483647)MAX ACCESS not accessibleSTATUScurrentDESCRIPTION"The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job specific objects. This value shall be the same for a job in the jmQueueTable as the corresponding jmJobIndex value in the jmJobTable for this job.The value 0 shall not be generated. Agents instrumenting
1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495 1496	JmCompletedJobIndex OBJECT TYPESYNTAX Integer32(12147483647)MAX ACCESS not accessibleSTATUS currentDESCRIPTION"The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job specific objects. This value shall be the same for a job in the jmQueueTable as the corresponding jmJobIndex value in the jmJobTable for this job.The value 0 shall not be generated. Agents instrumenting systems that contain jobs with a job identifier of 0 shall map
1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495 1496 1497	<pre>JMCOMPletedJobIndex OBJECT TYPE SYNTAX Integer32(12147483647) MAX ACCESS not accessible STATUS current DESCRIPTION "The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job specific objects. This value shall be the same for a job in the jmQueueTable as the corresponding jmJobIndex value in the jmJobTable for this job. The value 0 shall not be generated. Agents instrumenting systems that contain jobs with a job identifier of 0 shall map the value 0 to a value that is one higher than the highest job</pre>
1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495 1496 1497 1498	<pre>JMCOMPletedJobIndex OBJECT TYPE SYNTAX Integer32(12147483647) MAX ACCESS not accessible STATUS current DESCRIPTION "The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job specific objects. This value shall be the same for a job in the jmQueueTable as the corresponding jmJobIndex value in the jmJobTable for this job. The value 0 shall not be generated. Agents instrumenting systems that contain jobs with a job identifier of 0 shall map the value 0 to a value that is one higher than the highest job identifier value that any job can have on that system."</pre>
1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495 1496 1497 1498 1499	<pre>JMCOMPletedJobIndex OBJECT TYPE SYNTAX Integer32(12147483647) MAX ACCESS not accessible STATUS current DESCRIPTION "The job's identifier generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job specific objects. This value shall be the same for a job in the jmQueueTable as the corresponding jmJobIndex value in the jmJobTable for this job. The value 0 shall not be generated. Agents instrumenting systems that contain jobs with a job identifier of 0 shall map the value 0 to a value that is one higher than the highest job identifier value that any job can have on that system." ;;= [jmCompletedEntry 2]</pre>

1501	
1502	
1503	
1504	The Job ID Group (Mandatory)
1505	
1500	The JEJODIDGroup consists entirely of the JEJODIDTADLE.
1507	${}$
1508	im Tobset Index and im TobIndex are materialized in this group
1510	
1511	Implementation of every object in this group is mandatory.
1512	See Section 4 entitled 'Conformance Considerations' on page 26.
1513	
1514	jmJobID OBJECT IDENTIFIER ::= { jobmonmib 6 }
1515	
1516	jmJobIDTable OBJECT-TYPE
1517	SYNTAX SEQUENCE OF JmJobIDEntry
1518	MAX-ACCESS not-accessible
1519	STATUS current
1520	DESCRIPTION
1521	submiggion ID that a glient upog to refer to a job and the
1522	im TobSet Index and im TobIndex that the Job Monitoring MIR agent
1525	assigned to the job and that is used to access the job in all of
1525	the other tables in the MIR. If a monitoring application
1526	already knows the imJobIndex of the job it is guerving, that
1527	application need not use the jmJobIDTable .
1528	
1529	See Terminology and Job Model on page 10 for the definition of a
1530	job set.
1531	
1532	The jmJobIDTable is indexed by:
1533	
1534	1. JmJobSubmissionIDIndex - a 32-octet job identifier
1535	generated when the job was submitted, either by the citent
1530	$\cdots = \int \frac{dr}{dr} \frac{dr}{dr$
1538	
1539	imJobIDEntry OBJECT-TYPE
1540	SYNTAX JmJobIDEntry
1541	MAX-ACCESS not-accessible
1542	STATUS current
1543	DESCRIPTION
1544	"The map from (1) the jmJobSubmissionIDIndex to (2) the
1545	jmJobSet Id Index and jmJobIndex.
1546	
1547	An entry shall exist in this table for each job, no matter what
1348 1540	Lie state of the job and no matter what job set the job is in.
1549	TNDEX / imTobSubmissionTDIndex
1550	$::= \{ jmJobJDTable 1 \}$
1552	
1553	JmJobIDEntry ::= SEQUENCE {

JMJODSUDMISSIONIDINDEX	OCTET STRING(SIZE(063)),
jmJob 1D SetIndex	Integer32(12147483647),
Jm JODID JODINdex	Integer32(12147483647),
	_
JMJODSUDMISSIONIDINGEX OBJECT-TYP.	
SYNTAX OCTET STRING(SIZE	(063))
MAX-ACCESS not-accessible	
STATUS current	
DESCRIPTION	
<u>"A quasi-unique string ID</u>	which identifies the job uniquely
within a particular clien	t-server environment. Either the
client or the server assig	gns the job submission iD for each job
ine monitoring application	n whether in the client or running
separately, uses the job	submission ID to help the user identify
which jmJobIndex was assig	gned by the agent.
There are multiple format	s tor the
<u>jmJobSubmissionCurrent</u> IDI	ndex. Each format shall be registered
using the procedures of a	type 2 enum. See section entitled:
<u> </u> IANA Registration of enu	ms <u>' on page 28.</u>
The value of jmJobSubmiss	ionIDIndex should be one of the
registered format types.	The first two octets of the string
shall indicate which regi	stered format is being used. The agent
shall assign a string of :	registered format (00) for any job
without a value. The form	mat values registered so far are:
Format	
Number Description	
00 Set by the age	nt when neither the client nor the
server_assigned	d a job submission ID.
01 octets 3-10:	8-decimal-digit random number
octets 11-32:	last 22 bytes of the jobName attribute
02 octets 3-10:	8-decimal-digit sequential number
octets 11-32: 0	Client MAC address
03 octets 3-10:	8-decimal-digit sequential number
octets 11-32:	last 22 bytes of the client URL
04 to be register	ed according to procedures of a type 2
enum.	
NOTE - the job submission	id only intended to be unique between
a limited set of clients	for a limited duration of time, namely
for the life time of the	job in the context of the server or
device that is processing	the job. Some of the formats include
something that is unique	per client and a random number so that
the same job submitted by	the same client will have a different
ioh gubmiggion id For o	ther formats where part of the id is

[Page 78]

	Job Monitoring MIB, V0. <u>871</u> April 4, 1997
1604 1605 1606 1607 1608 1609	guaranteed to be unique for each client, 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 been selected to reduce the probability of collision to a very low number, but is not intended to be an absolute guarantee of uniqueness. None-the-
1610 1611 1612 1613 1614	<pre>less, collisions could occur, but without bad consequences, since this MIB is intended to be used only for monitoring jobs, not for controlling and managing them." ::= { jmJobIDEntry 1 }</pre>
1615 1616 1617	jmJobHDSetIndex OBJECT-TYPE SYNTAX Integer32(12147483647) MAX-ACCESS read-only
1618 1619 1620	STATUS current DESCRIPTION "The job set index of the job set in which the job was placed
1621 1622 1623 1624	when that server or device accepted the job. This value in combination with the jmEdJobIndex value permits the management application to access the other tables to obtain the job- specific objects. This value shall be the same for a job in the
1625 1626 1627 1628 1629	<u>jmJobStateTable</u> and <u>jmAttributeTable</u> for this job. <u>NOTE - an implementation that has only one job set, such as a</u> <u>printer with a single queue shall hard code this object with</u>
1629 1630 1631 1632 1633	the value 1. See Terminology and Job Model on page 10 for the definition of a job set." ::= { jmJobIDEntry 2 }
1634 1635 1636	jm ID JobIndex OBJECT-TYPE SYNTAX Integer32(12147483647) MAX-ACCESS read-only
1637 1638 1639	STATUS current DESCRIPTION "The sequential, monatonically increasing identifier for the job
1640 1641 1642 1643 1644	generated by the server or device when that server or device accepted the job. This value permits the management application to access the other tables to obtain the job-specific objects. This value shall be the same for a job in the jmJobSubmissionIDTable as the corresponding jmJobIndex value in
1645 1646 1647	the jmJobStateTable and jmAttributeTable for this job. The value 0 shall <i>not</i> be generated. Agents instrumenting
1648 1649 1650 1651 1652	<pre>systems that contain jobs with a job identifier of 0 shall map the value 0 to a value that is one higher than the highest job identifier value that any job can have on that system." ::= { jmJobIDEntry 3 }</pre>
1653 1654 1655 1656	The Job <u>State</u> Group (Mandatory)

```
1657
1658
      -- The jmJobStateGroup consists entirely of the jmJobStateTable.
1659
      --
1660
      -- Implementation of every object in this group is mandatory.
      -- See Section 4 entitled 'Conformance Considerations' on page 26.
1661
1662
1663
      jmJobState OBJECT IDENTIFIER ::= { jobmonmib 78 }
1664
1665
      jmJobStateTable OBJECT-TYPE
1666
          SYNTAX
                       SEQUENCE OF JmJobStateEntry
1667
          MAX-ACCESS not-accessible
1668
          STATUS
                       current
          DESCRIPTION
1669
1670
               "The jmJobStateTable consists of basic job state<del>identification</del>
1671
               and status information for each job in a job set that (1)
               monitoring applications need to be able to access in a single
1672
1673
               SNMP Get operation, (2) that have a single value per job, and
1674
               (3) that shall always be implemented. See Terminology and Job
              Model on page 10 for the definition of a job set.
1675
1676
1677
              NOTE - Every accessible object in this table shall have the same
1678
               value as one of the attributes in the jmAttributeTable.
               Implementations may either keep a separate copy or may share
each value that is common between the jmJobStateTable and the
1679
1680
1681
               jmAttributeTable. The persistence of the two tables may be
1682
              different depending on implementation and/or system
               administrator policy as specified by the jmGeneralJobPersistence
1683
1684
               and jmGeneralAttributePersistence objects defined on page 69.
               Thus an accounting application need only copy the entire
1685
               jmAttributeTable or selected job rows and will obtain all of the information about the job and its state.
1686
1687
1688
1689
               The jmJobStateTable is indexed by:
1690
1691
               1.
                   jmJobSetIndex - a running index of Job Set instances
                   supported by this device or server. A job set is used in
1692
1693
                   the MIB to represent the separation of jobs into disjoint
                   sets for scheduling purposes in a server, typically into
1694
                   separate job queues. See Terminology and Job Model on page
1695
1696
                   10 for the definition of a job set.
1697
1698
                   jmJobIndex - the job identifier that was generated by the
               2.
1699
                   server or device that accepted the job."
1700
          ::= { jmJobState 1 }
1701
1702
      jmJobStateEntry OBJECT-TYPE
1703
          SYNTAX
                       JmJobStateEntry
1704
          MAX-ACCESS not-accessible
1705
          STATUS
                       current
1706
          DESCRIPTION
1707
               "Basic per-job state<del>identification</del> and status information.
1708
```

1709 1710 1711 1712 1713 1714	An entry shall exist in this table the state of the job is. Each job one job set." INDEX { jmJobSetIndex , jmJobIndex } ::= { jmJob <mark>State</mark> Table 1 }	e for each job, no matter what o shall appear in one and only
1715	JmJob <mark>State</mark> Entry ::= SEQUENCE {	
1716	- Job Identification (I) objects:	
	jmJobIndex	Integer32(12147483647),
	jmJobName	OCTET STRING(SIZE(063)),
	jmJobIdName	OCTET STRING(SIZE(063)),
	jmJobIdNumber	Integer32(02147483647),
	JmJobServiceTypes	Integer32(1214/48364/),
	in Joh Com on	JIJODSETVICETYPESTC
	JIIJODUWIICI im JohDowi goNomoOrOuouoDoguogtod	$\frac{\text{OCTET STRING(SIZE(003))}}{\text{OCTET CUDINC(CIZE(003))}}$
	jmJobGurrentState	Jm JobStateTC
	im TobStateReagong	$\frac{OCTET STRING(STZE(0 - 62))}{OCTET STRING(STZE(0 - 62))}$
	Jub obb catericaborib	encoded as a bit string
	jmJobStateKOctetsCompleted	Integer $32(0.2147483647)$,
	imJobStateImpressionsCompleted	Integer32(02147483647),
	jmJobStateAssociatedValue	Integer32(02147483647)
1717	}	
1718		
	— The following jmJobGroup objects ident — the management application which may b — end user or a system operator:	ify the job to the user of e acting in the role of an
1720		
1721	jmJobIndex OBJECT TYPE	
1722	SYNTAX Integer32(12147483647)	
1723	MAX ACCESS not accessible	
1724	STATUS current	
1725	DESCRIPTION	
1/20	"The identifier of the job on the	device or server. The job's
1720	identifier is generated by the set	rver or device when that server
1720	generate a job identifier for each	ver, if the device does not
1729	MIP agent shall generate the job	dontifior for the job
1730	Mib agent sharr generate the job.	idencifier for the job.
1732	The value 0 shall not be generated	Agenta instrumenting
1733	systems that contain jobs with a	iob identifier of 0 shall map
1734	the value 0 to a value that is one	higher than the highest job
1735	identifier value that any job can	have on that system."
1736	++= { jmJobEntry 1 }	4 1 1 1
1737		
1738	jmJobName OBJECT TYPE	
1739	SYNTAX OCTET STRING(SIZE(063))	
1740	MAX ACCESS read only	
1741	STATUS current	
1742	DESCRIPTION	

1743	"This object is the human readable string name of the job as
1744	assigned by the submitting user to help the user distinguish
1745	between his/her various jobs. This name does not need to be
1746	unique.
1747	
1748	This attribute is intended for enabling a user or the user's
1749	application to convey a job name that may be printed on a start
1750	sheet. returned in a guery result. or used in notification or
1751	logging messages.
1752	
1753	If this attribute is not specified when the job is submitted no
1754	ich name is assumed but implementation specific defaults are
1755	allowed auch as the value of the desumentName(4) resource item
1756	of the first document in the job or the fileName(2) resource
1750	item of the finat document in the job of the lifewalle()
1750	
1750	mine descriptions in the line in the large for the description of the second second second second second second
1759	The JmJobName is distinguished from the JobComment attribute, in
1760	that the JmJobName is intended to permit the submitting user to
1/61	distinguish between different jobs that he/she has submitted.
1762	The jobComment attribute is intended to be free form additional
1763	information that a user might wish to use to communicate with
1764	himself/herself, such as a reminder of what to do with the
1765	results or to indicate a different set of input parameters were
1766	tried in several different job submissions."
1767	··= { jmJobEntry 2 }
1768	
1769	jmJobIdName OBJECT TYPE
1769 1770	jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063))
1769 1770 1771	jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only
1769 1770 1771 1772	jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current
1769 1770 1771 1772 1773	jmJobidName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION
1769 1770 1771 1772 1773 1774	jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process
1769 1770 1771 1772 1773 1774 1775	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the</pre>
1769 1770 1771 1772 1773 1774 1775 1776	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object.</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777	jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object.
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778	jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The imJobIdName and the imJobIdNumber objects are referred to as
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user.</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator. or the system administrator to uniquely identify the</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print_jobg_of_interest_from_all_the_jobg_currently_"known" by</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1781	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the gerver or device.</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device.</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1783	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device.</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1783	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job current is a comparison of the system of a comparison of the server.</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1784	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the sigh submission protocol</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1784 1785 1786	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the job submission protocol. The format of the goded abaracter act data or an interest.</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1784 1785 1786 1787	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client-side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the job submission protocol. The format of the coded character set data down down of the server. </pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1786 1787	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the job submission protocol. The format of the coded character set data and point of assignment of the client side identifiers depend upon the job</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1786 1787 1788 1789	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the job submission protocol. The format of the coded character set data and point of assignment of the client side identifiers depend upon the job submission protocol in use. See Appendix A on page for the</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1786 1787 1788 1789 1790	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the job submission protocol. The format of the coded character set data and point of assignment of the client side identifiers depend upon the job submission protocol in use. See Appendix A on page for the mapping from selected job submission protocols to these client-</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1786 1787 1788 1789 1790 1791	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the job submission protocol. The format of the coded character set data and point of assignment of the client side identifiers depend upon the job submission protocol in use. See Appendix A on page for the mapping from selected job submission protocols to these client side job identifiers. </pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1788 1785 1786 1787 1788 1789 1790 1791 1792	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the job submission protocol. The format of the coded character set data and point of assignment of the client side identifiers depend upon the job submission protocol in use. See Appendix A on page for the mapping from selected job submission protocols to these client- side job identifiers.</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1788 1785 1786 1787 1788 1789 1790 1791 1792 1793	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX ACCESS read only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the job submission protocol. The format of the coded character set data and point of assignment of the client side identifiers depend upon the job submission protocol in use. See Appendix A on page for the mapping from selected job submission protocols to these client- side job identifiers. Unlike jmJobName, which is assigned by the submitting user, the</pre>
1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1788 1785 1786 1787 1788 1789 1790 1791 1792 1793 1794	<pre>jmJobIdName OBJECT TYPE SYNTAX OCTET STRING(SIZE(063)) MAX_ACCESS read-only STATUS current DESCRIPTION "Identifies the job on the "client side" of the printing process as coded character set data in combination with the jmJobIdNumber object. The jmJobIdName and the jmJobIdNumber objects are referred to as the "client side" identifiers because they allow the user, operator, or the system administrator to uniquely identify the print jobs of interest from all the jobs currently "known" by the server or device. The client side identifiers can be assigned by either the job submission client's local system or a downstream server, depending on implementation and the job submission protocol. The format of the coded character set data and point of assignment of the client side identifiers depend upon the job submission protocol in use. See Appendix A on page for the mapping from selected job submission protocols to these client- side job identifiers. Unlike jmJobName, which is assigned by the submitting user, the jmJobIdName and jmJobIdNumber client side identifiers provide</pre>

1796	
1797	The imJobIdName object may be used alone or in conjunction with
1798	the imJobIdNumber object, depending upon the format of the job
1799	submission protocol client side identifier. For example, the
1800	LPD job identifier normally contains three alpha characters
1801	followed by a three digit number. The agent may represent the
1802	alpha portion by imJobIdName and the numeric portion by
1803	im Job Id Number. Alternatively, the agent may represent the LPD
1804	client side id entirely in the im.TobIdName object."
1805	\div
1806	
1807	imJobIdNumber OBJECT TYPE
1808	SYNTAX Integer32(02147483647)
1809	MAX-ACCESS read only
1810	STATUS current
1811	DESCRIPTION
1812	"Identifies the job on the "client side" of the printing process
1813	in combination with the jmJobIdName object. This object may be
1814	used alone or in conjunction with the jmJobIdName object,
1815	depending upon the format of the job submission protocol client-
1816	side identifier. Refer to the jmJobIdName object specification.
1817	
1818	If the value of this object is unknown, the agent shall return
1819	the value (-2)."
1820	··= { jmJobEntry 4 }
1821	
1000	
1022	JUDDBELVICETYPES ODDECT TITE
1822	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on
1822 1823 1824	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page
1822 1823 1824 1825	STREET TIPE
1822 1823 1824 1825 1826	SYNTAX Integer32(12147483647) SYNTAX Integer32(12147483647) page MAX ACCESS read only STATUS current
1822 1823 1824 1825 1826 1827	STATUS SYNTAX Integer32(12147483647) SYNTAX Integer32(12147483647) page MAX ACCESS read only STATUS current DESCRIPTION
1822 1823 1824 1825 1826 1827 1828	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page MAX ACCESS read only STATUS current DESCRIPTION "Specifies the type(s) of service to which the job has been
1822 1823 1824 1825 1826 1827 1828 1829	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page MAX ACCESS read only STATUS current DESCRIPTION "Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is
1822 1823 1824 1825 1826 1827 1828 1829 1830	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page MAX ACCESS read only STATUS current 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
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read onlySTATUS currentDESCRIPTION"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,
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read onlySTATUSCurrentDESCRIPTION"Specifies the type(s) of service to which the job has beensubmitted (print, fax, scan, etc.). The service type isrepresented as an enum that is bit encoded with each job servicetype so that more general and arbitrary services can be created,such as services with more than one destination type, or ones
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read onlySTATUSCurrentDESCRIPTION"Specifies the type(s) of service to which the job has beensubmitted (print, fax, scan, etc.). The service type isrepresented as an enum that is bit encoded with each job servicetype so that more general and arbitrary services can be created,such as services with more than one destination type, or oneswith only a source or only a destination. For example, a job
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read onlySTATUS currentDESCRIPTION"Specifies the type(s) of service to which the job has beensubmitted (print, fax, scan, etc.). The service type isrepresented as an enum that is bit encoded with each job servicetype so that more general and arbitrary services can be created,such as services with more than one destination type, or oneswith only a source or only a destination. For example, a jobservice might scan, fax, and print a single job. In this case,
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read onlySTATUS currentDESCRIPTION"Specifies the type(s) of service to which the job has beensubmitted (print, fax, scan, etc.). The service type isrepresented as an enum that is bit encoded with each job servicetype so that more general and arbitrary services can be created,such as services with more than one destination type, or oneswith only a source or only a destination. For example, a jobservice might scan, fax, and print a single job. In this case,three bits would be set in the jmJobServiceTypes object,
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read onlySTATUS currentDESCRIPTION"Specifies the type(s) of service to which the job has beensubmitted (print, fax, scan, etc.). The service type isrepresented as an enum that is bit encoded with each job servicetype so that more general and arbitrary services can be created,such as services with more than one destination type, or oneswith only a source or only a destination. For example, a jobservice might scan, fax, and print a single job. In this case,three bits would be set in the jmJobServiceTypes object,corresponding to the values: 8+32+4=44, respectively.
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read onlySTATUS currentDESCRIPTION"Specifies the type(s) of service to which the job has beensubmitted (print, fax, scan, etc.). The service type isrepresented as an enum that is bit encoded with each job servicetype so that more general and arbitrary services can be created,such as services with more than one destination type, or oneswith only a source or only a destination. For example, a jobservice might scan, fax, and print a single job. In this case,three bits would be set in the jmJobServiceTypes object,corresponding to the values: 8+32+4=44, respectively.
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read onlySTATUS currentDESCRIPTION"Specifies the type(s) of service to which the job has beensubmitted (print, fax, scan, etc.). The service type isrepresented as an enum that is bit encoded with each job servicetype so that more general and arbitrary services can be created,such as services with more than one destination type, or oneswith only a source or only a destination. For example, a jobservice might scan, fax, and print a single job. In this case,three bits would be set in the jmJobServiceTypes object,corresponding to the values: 8+32+4=44, respectively.
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1849	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read only STATUS currentDESCRIPTION"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, fax, and print a single job. In this case, three bits would be set in the jmJobServiceTypes object, corresponding to the values: 8+32+4=44, respectively.Whether this object is set from a job attribute supplied by the job submission client or is set by the recipient job submission
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1840	<pre>SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page MAX ACCESS read only STATUS current 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, fax, and print a single job. In this case, three bits would be set in the jmJobServiceTypes object, corresponding to the values: 8+32+4=44, respectively.</pre>
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1841	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on pageMAX ACCESS read only STATUS currentDESCRIPTION"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, fax, and print a single job. In this case, three bits would be set in the jmJobServiceTypes object, corresponding to the values: 8+32+4=44, respectively.Whether this object 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
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842	<pre>SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page MAX ACCESS read only STATUS current 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, fax, and print a single job. In this case, three bits would be set in the jmJobServiceTypes object, corresponding to the values: 8+32+4=44, respectively. Whether this object 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 object indicating the type of the job.</pre>
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842 1843	SYNTAX Integer32(12147483647)See JmJobServiceTypesTC onpageMAX ACCESS read onlySTATUS currentDESCRIPTION"Specifies the type(s) of service to which the job has beensubmitted (print, fax, scan, etc.). The service type isrepresented as an enum that is bit encoded with each job servicetype so that more general and arbitrary services can be created,such as services with more than one destination type, or oneswith only a source or only a destination. For example, a jobservice might scan, fax, and print a single job. In this case,three bits would be set in the jmJobServiceTypes object,corresponding to the values: 8+32+4=44, respectively.Whether this object is set from a job attribute supplied by thejob submission client or is set by the recipient job submissionserver or device depends on the job submission protocol. Withother indicating the type of the job.
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845	<pre>SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page MAX ACCESS read only STATUS current 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, fax, and print a single job. In this case, three bits would be set in the jmJobServiceTypes object, corresponding to the values: 8+32+4=44, respectively. Whether this object 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 object indicating the type of the job. One of the purposes of this object is to permit a requester to return the purposes of this object is to permit a requester to one of the purposes of this object is to permit a requester to return the purposes of this object is to permit a requester to return the purposes of this object is to permit a requester to return the purposes of this object is to permit a requester to return the purposes of this object is to permit a requester to return the purposes of this object is to permit a requester to return the purposes of this object is to permit a requester to return the purposes of this object is to permit a requester to return the purposes of this object is to permit a requester to return the permit of the purposes of the permit of the permit</pre>
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845	<pre>SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page MAX ACCESS read only STATUS current 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, fax, and print a single job. In this case, three bits would be set in the jmJobServiceTypes object, corresponding to the values: 8+32+4=44, respectively. Whether this object 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 object indicating the type of the job. One of the purposes of this object is to permit a requester to filter out jobs that are not of interest. For example, a</pre>
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846	<pre>SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page MAX ACCESS read only STATUS current 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, fax, and print a single job. In this case, three bits would be set in the jmJobServiceTypes object, corresponding to the values: 8+32+4=44, respectively. Whether this object 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 object indicating the type of the job. One of the purposes of this object 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</pre>
1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846 1847	<pre>SYNTAX Integer32(12147483647)See JmJobServiceTypesTC on page MAX ACCESS read only STATUS current 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, fax, and print a single job. In this case, three bits would be set in the jmJobServiceTypes object, corresponding to the values: 8+32+4=44, respectively. Whether this object 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 object indicating the type of the job. One of the purposes of this object 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 object is in the job identification</pre>

18/10	
1049	This object is a type 0 enum
1050	IIIIS ODJECT IS a type z enum.
1851	
1852	The JmJobServiceTypesTC textual convention defines component
1853	types as separate bit value in the enum. See page ."
1854	++= { jmJobEntry 5 }
1855	
1856	imJobOwner OBJECT TYPE
1857	SYNTAX OCTET STRING(SIZE(063))
1858	MAX ACCESS read only
1859	STATUS gurrent
1860	DECONDUCIÓN
1861	"The goded abarrator get name of the upper that submitted the
1001	The coded character set hame of the user that submitted the
1862	Job. The method of assigning this user name will be system
1863	and/or site specific but the method must insure that the name is
1864	unique to the network that is visible to the client and target
1865	device.
1866	
1867	This value should be the authenticated name of the user
1868	submitting the job."
1869	+ = { imJobEntry 6 }
1870	
1871	
1872	
1872	MAX ACCESS road only
1075	
10/4	DECONDUCIÓN
18/5	DESCRIPTION
18/6	"The administratively defined coded character set name of the
18/7	target device or queue. Its value corresponds to the Printer
1878	MIB: prtGeneralAdminName object (added to the draft Printer MIB)
1879	for printers. For servers, this object is the name that users
1880	supply to indicate whether they want the job to be processed,
1881	typically, but not limited to, a job queue name or logical
1882	printer name."
1883	\div = $\frac{1}{10000000000000000000000000000000000$
1884	
1885	imJobCurrentState OBJECT-TYPE
1886	SVNTAX JIM JobStateTC See page 32
1887	MAX_ACCESS read_only
1007	
1000	
1000	DESCRIPTION
1890	"The current state of the job (pending , processing , held , etc.).
1891	
1892	The value of this object shall always be the same as that of the
1893	jobState attribute, so that this information appears in both the
1894	jmJobStateTable and the jmAttributeTable simultaneously. See
1895	the JmJobStateTC textual-convention on page 32 and the jobState
1896	attribute on page 39 in the jmAttributeTable for the full
1897	specification of this object/attribute.Management applications
1898	shall be prepared to receive all the standard job states.
1899	Servers and devices are not required to generate all job states
1900	only those which are appropriate for the particular
1901	implementation
1/01	

1902	
1903	A companion textual convention (JmJobStateReasonsTC) and
1904	corresponding object (imJobStateReasons) provide additional
1905	information about job states. While the job states cannot be
1906	added to without impacting deployed clients, it is the intent
1907	that additional JmJobStateReasonsTC enums can be defined without
1908	impacting deployed clients. In other words, the
1909	Impacting appropria difference. In concer words, one
1910	Chooped Concepter is incended to be encendible. See page .
1911	This object is a type 2 enum "
1912	$::= \{ imJobStateEntry 1 + \}$
1913	
1914	imJobStateReasons OBJECT TYPE
1915	SYNTAX OCTET STRING(SIZE(063)) encoded as a bit string
1916	
1917	
1918	MAX ACCESS read only
1919	STATUS durrent
1920	DECOLDTION
1020	"This object provides additional information resarding the
1022	im Top Current State object This object identifies the reason or
1922	reagong that the job is in the propresenting hold pending
1923	progogging poodelttention payeed interrupted terminating
1924	processing, needsActencion, paused, incertupted, terminating,
1925	recarried, or completed state. The server shall indicate the
1920	in TebCheteDesgeng object. While the jeb states sonnet be added
1927	JIDODStateReasons object. While the job states cannot be added
1928	to without impacting deproyed clients, it is the intent that
1929	additional JEJODStateReasonsit enums can be defined without
1930	Impacting deployed cilents. In other words, the
1931	JEJODStatekeasonsic is intended to be extensible. See page .
1932	
1933	when the job does not have any reasons for being in its current
1934	state, the server shall set the value of the JmJobStateReasons
1935	object to a bit string containing all zeros.
1936	
1937	Bits in the bit string are assigned starting with the most
1938	significant bit in the most significant octet which is called
1939	bit 1. Bit 2 is the next most significant bit in the most
1940	significant octet, etc. Bit 9 is the most significant bit in
1941	the second most significant octet, etc., up to the maximum bit:
1942	504 (= 8 x 63). See JmJobStateReasonsTC on page
1943	
1944	An agent only need return the most significant octet up to the
1945	least significant octet that contains a non-zero bit.
1946	
1947	If all bits are zero, the agent may return an OCTET STRING of
1948	zero length. Alternatively, an agent may always return a fixed
1949	number of octets starting with the most significant octet and
1950	running through the least significant octet that could ever have
1951	a one bit in it for that implementation.
1952	
1953	This object is a type 2 bit string. See Section "
1954	::= { jmJobEntry 9 }

<pre>MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of octets completed processing by the or device measured in units of K (1024) octets. The value of this object shall always be the same as that jobKoctetsCompleted attribute, so that this information a in both the jmJobStateTable and the jmAttributeTable simultaneously. See the jobKoctetsCompleted attribute or 49 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 2 } mJobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute c 50 in the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute c 50 so far. The value of the most relevant attribute associated with jobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the inpressionsCompleted attribute. See the inpressionsCompleted attribute. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the inpressionsCompleted attributeTable so that this informati appears in both the informatice. See the info</pre>
STATUS current DESCRIPTION "The current number of octets completed processing by the or device measured in units of K (1024) octets. The value of this object shall always be the same as that jobKoctetsCompleted attribute, so that this information a in both the jmJobStateTable and the jmAttributeTable simultaneously. See the jobKoctetsCompleted attribute on 49 in the jmAttributeTable for the full specification of object/attribute." ::= [jmJobStateEntry 2] mJobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of object/attribute." :::= { jmJobStateEntry 3 } mJobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simUtaneously. See the inpStateTable and the jmAttributeTable
DESCRIPTION "The current number of octets completed processing by the or device measured in units of K (1024) octets. The value of this object shall always be the same as that jobKoctetsCompleted attribute, so that this information a in both the jmJobStateTable and the jmAttributeTable simultaneously. See the jobKOctetsCompleted attribute or 49 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 2 } JobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that igbStateAssociatedValue attribute, so that this informatian if the value of this object shall always be the same as that jobStateAssociatedValue OBJECT-TYPE STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informatian appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the inbStateTable and the jmAttribute associatedValue attribute, so that this informatian appears in both the jmJobStateTable and the jmAttribute
<pre>"The current number of octets completed processing by the or device measured in units of K (1024) octets. The value of this object shall always be the same as that jobKoctetsCompleted attribute, so that this information a in both the jmJobStateTable and the jmAttributeTable simultaneously. See the jobKoctetsCompleted attribute or 49 in the jmAttributeTable for the full specification of object/attribute." ::= [jmJobStateEntry 2] JobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute o 50 in the jmAttributeTable for the full specification of object/attribute." ::= [jmJobStateEntry 3] JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informatian in both the jmJobStateTable and the jmAttribute associated with job's current bate. The value of the most relevant attribute associated with job's current state.</pre>
<pre>or device measured in units of K (1024) octets. The value of this object shall always be the same as that jobKOctetsCompleted attribute, so that this information a in both the jmJobStateTable and the jmAttributeTable simultaneously. See the jobKOctetsCompleted attribute or 49 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 2 } JobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute o fo in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute o solvet/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that iobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the abStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the abStateTable and the jmAttributeTable simultaneously.</pre>
The value of this object shall always be the same as that jobKOctetsCompleted attribute, so that this information a in both the jmJobStateTable and the jmAttributeTable simultaneously. See the jobKOctetsCompleted attribute or 49 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 2 } JobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informatian appears in both the jmJobStateTable and the jmAttributeTable SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informatian appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informatian appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informatian appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informatian appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informatian appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See
<pre>jobKOctetsCompleted attribute, so that this information a in both the jmJobStateTable and the jmAttributeTable simultaneously. See the jobKOctetsCompleted attribute or 49 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 2 } JobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 00 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informatic appears in both the jmJobStateTable and the jmAttributeTable information object/attribute." ::= The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informatic appears in both the jmJobStateTable and the jmAttributeTable and the jmAttributeTable attribute.</pre>
<pre>in both the jmJobStateTable and the jmAttributeTable simultaneously. See the jobKOctetsCompleted attribute or 49 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 2 } mJobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of object/attribute." ::= { jmJobStateEntry 3 } mJobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati jobStateAssociatedValue attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informaties appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informaties appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informaties appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informaties appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informaties appears in both the jmJobStateTable and the jmAttributeTable simul</pre>
<pre>simultaneously. See the jobKOctetsCompleted attribute or 49 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 2 } JobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute c 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informatile and state at the simultaneously. Status current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the informatile appears in both the jmJobStateTable and the jmAttributeTa stributeTable simultaneously. See the informatile appears in both the jmJobStateTable and the jmAttributeTable stributeTable simultaneously. See the informatile appears in both the jmJobStateTable and the jmAttributeTable stributeTable simultaneously. See the informatile appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the informatile appears in the simultaneously. See the informatile appears in the simultaneously. See the informatile appears in the simultaneously. See the informat</pre>
<pre>49 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 2 } JobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the intervant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable asimultaneously. See the intervant attribute attribute attribute associated value attribute.</pre>
<pre>object/attribute." ::= { jmJobStateEntry 2 } fobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute c 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } fobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable ad the jmAttributeTable simultaneously. See the imJobStateTable at the same as that intervalue of the imJobStateTable and the jmAttributeTable as impltaneously. See the imJobStateTable and the jmAttributeTable as the imJobStateTable and the jmAttributeTable and the jmAttributeTable as the imJobStateTable and the jmAttributeTable as the imJobStateTable and the jmJobStateTable a</pre>
<pre>::= { jmJobStateEntry 2 } JobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION</pre>
TobStateImpressionsCompleted OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of object/attribute." :::= { jmJobStateEntry 3 } TobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable attribute associated with job's current state.
<pre>SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute c 50 in the jmAttributeTable for the full specification of object/attribute." ::= [jmJobStateEntry 3] JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted with processions implement to be appear attribute.</pre>
<pre>MAX-ACCESS read-only STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTable simultaneously. See the inpStateTable and the jmAttributeTable simultaneously See the inpStateTable and the jmAttributeTable and the jmAttributeTable and the jmAttributeTable and the jmAttributeTable attributeTable and the jmAttributeTable and</pre>
STATUS current DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable contribute
DESCRIPTION "The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the inDStateAssociatedValue attribute
<pre>"The current number of impressions completed being marked stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the iobStateAssociatedValue attribute</pre>
<pre>stacked by the device for this job so far. The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the inbStateAssociatedValue attribute</pre>
The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the inbStateAssociatedValue attribute
The value of this object shall always be the same as that impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute of 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the inpStateAssociatedValue attribute
<pre>impressionsCompleted attribute, so that this information in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute c 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the iobStateAssociatedValue attribute</pre>
<pre>in both the jmJobStateTable and the jmAttributeTable simultaneously. See the impressionsCompleted attribute c 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the jobStateAssociatedValue attribute</pre>
<pre>simultaneously. See the impressionsCompleted attribute c 50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the jobStateAssociatedValue attribute</pre>
<pre>50 in the jmAttributeTable for the full specification of object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the jobStateAssociatedValue attribute</pre>
<pre>object/attribute." ::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the jobStateAssociatedValue attribute</pre>
<pre>::= { jmJobStateEntry 3 } JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the jobStateAssociatedValue attribute</pre>
JobStateAssociatedValue OBJECT-TYPE SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously.
SYNTAX Integer32(02147483647) MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously
MAX-ACCESS read-only STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the jobStateAssociatedValue attribute
STATUS current DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously.
DESCRIPTION "The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the jobStateAssociatedValue attribute
"The value of the most relevant attribute associated with job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa
job's current state. The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the jobStateAssociatedValue attribut
The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa
The value of this object shall always be the same as that jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa simultaneously. See the jobStateAssociatedValue attribut
jobStateAssociatedValue attribute, so that this informati appears in both the jmJobStateTable and the jmAttributeTa
appears in both the jmJobStateTable and the jmAttributeTa
simultaneously See the inhStateAccoriatedValue attribut
simulaneously. See the JobstateAssociatedvalue attribut
page 39 in the jmAttributeTable for the full specification
this object/attribute."
::= { jmJobStateEntry 4 }

2005 2006 2007 -- The Attribute Group (Mandatory) 2008 2009 -- The jmAttributeGroup consists entirely of the jmAttributeTable. 2010 _ _ 2011 -- Implementation of every object in this group is mandatory. -- See Section 4 entitled 'Conformance Considerations' on page 26. 2012 2013 _ _ 2014 -- Some attributes are mandatory for conformance, and the rest are 2015 -- optional. The mandatory attributes are the ones required to have -- copies in the jmJobStateTable. The mandatory attributes are: 2016 2017 _ _ 2018 ___ jobState numberOfInterveningJobs 2019 ___ 2020 deviceAlertCode ___ 2021 jobKOctetsRequestedTotal --2022 jobKOctetsCompleted ___ 2023 ___ impressionsRequestedTotal impressionssheetsCompleted 2024 _ _ 2025 ___ outputBinNameIndex 2026 2027 2028 jmAttribute OBJECT IDENTIFIER ::= { jobmonmib 89 } 2029 2030 jmAttributeTable OBJECT-TYPE 2031 SEQUENCE OF JmAttributeEntry SYNTAX 2032 MAX-ACCESS not-accessible 2033 STATUS current 2034 DESCRIPTION 2035 "The **jmAttributeTable** <u>shall</u> contains attributes of the job and 2036 document(s) for each job in a job set. Instead of allocating 2037 distinct objects for each attribute, each attribute item-is 2038 represented as a separate row in the **jmAttributeTable**. Some 2039 attributes may represent information about the job and 2040 document(s), such as file-names, document-names, submission-2041 time, completion-time, size, etc. Other aAttributes may also 2042 represent requested and/or consumed resources for each job. 2043 2044 The **jmAttributeTable** is a per-job table with an extra index for each type of attribute (jmAttributeTypeIndex) that a job can 2045 have and an additional index (jmAttributeInstanceIndex) for 2046 2047 those attributes that can have multiple instances per job. The 2048 jmAttributeTypeIndex object shall contain an enum type that indicates the type of attribute (see **JmAttributeTypeTC** on page 2049 The value of the attribute shall be represented in either 2050 37). 2051 the jmAttributeValueAsInteger or jmAttributeValueAsOctets 2052 objects, or both, as specified in the JmAttributeTypeTC on page 2053 37. 2054 2055 The agent shall create rows in the jmAttributeTable as the 2056 server or printer is able to discover the attributes either from the job submission protocol itself or from the document PDL. 2057 As

2058 the documents are interpreted, the interpreter may discover 2059 additional attributes and so the agent adds additional rows to this table. As the resources are actually consumed, the usage 2060 2061 counter contained in the jmAttributeValueAsInteger object is 2062 incremented according to the units indicated in the description 2063 of the JmAttributeTypeTC enum. 2064 2065 The **jmAttributeTable** is indexed by (from most significant to 2066 least significant): 2067 2068 1) **jmJobSetIndex** - a running index of Job Set instances supported by this device or server. A job set is used in the 2069 2070 MIB to represent the separation of jobs into disjoint sets 2071 for scheduling purposes in a server, typically into separate 2072 job queues. See Terminology and Job Model on page 10 for the 2073 definition of a job set. 2074 2075 2) **jmJobIndex** - the job identifier that was generated by the 2076 server or device that accepted the job. 2077 2078 3) **jmAttributeTypeIndex** - the enum that indicates the type of 2079 the attribute. See JmAttributeTypeTC on page 37 for the 2080 specification of each attribute. 2081 2082 4) **jmAttributeInstanceIndex** - a running index of attributes of 2083 the same type for each job. For those attributes with only a 2084 single instance per job, this index value shall be 1. For 2085 those attributes that are a single value per document, the 2086 index value shall be the document number, starting with 1 for the first document in the job. Jobs with only a single document shall use the index value of 1. For those 2087 2088 2089 attributes that can have multiple values per job and per 2090 document, such as **documentFormatIndex** or **documentFormatEnum**, the index shall be a running index for the job as a whole, 2091 2092 starting at 1." 2093 ::= { jmAttribute 1 } 2094 2095 jmAttributeEntry OBJECT-TYPE 2096 SYNTAX JmAttributeEntry 2097 MAX-ACCESS not-accessible 2098 STATUS current 2099 DESCRIPTION 2100 "Attributes representing information about the job and 2101 document(s) or resources required and/or consumed. 2102 2103 Zero or more entries shall exist in this table for each job in a 2104 job set. Each job shall appear in one and only one job set." 2105 INDEX { jmJobSetIndex, jmJobIndex, jmAttributeTypeIndex, 2106 jmAttributeInstanceIndex } 2107 ::= { jmAttributeTable 1 } 2108 2109 JmAttributeEntry ::= SEQUENCE { jmAttributeTypeIndex JmAttributeTypeTC,

	jmAttributeInstanceIndex jmAttributeValueAsInteger imAttributeValueAsOctets	<pre>Integer32(132767), Integer32(02147483647), OCTET_STRING(SIZE(063))</pre>
2110	}	OCIEI SIRING(SIZE(U.:US))
2110	J	
2111	imattributeTureIndey ORIFOT_TVDF	
2112 2112		G_{00} mage 27
2113 2114	MAX ACCESC not accordible	see page si
2114	MAX-ACCESS HOU-accessible	
2115		
2110	UEBCRIFIION	
2117 2118	The type of attribute.	
2110	The time may identify inf	α
2119	The type may identify into	required to proceed the job of document(s)
2120	or may identify a resource	e required to process the job before
2121	the job start processing a	and/or consumed by the job as the job
2122	is processed.	
2123	There is a state and desure	ant information include.
2124	Examples of job and docume	ent information include.
2123	JODCOPIESREquested, docum	FileName and degumentName
2120	documentcoprescompreted,	LITENAME, and documentioname.
2127	Evernles of resources reg	vired and congumed include.
2120	ich Cateta Deguarted Tetal	
2129	JODKOCLELS <mark>Requested totat</mark> ,	JODKOCLELSCOMPIELEG, pageskequested,
2130	pagescompieted, mediumked	AttributeTreeTC textual convention on
2131	respectively. See the Jul	Actibule type it textual convention on
2132	page 37.	
2133	In the definitions of the	anyma in the Traltributerrand textual
2134 2125	appunction and descript	enums in the JMACTIDUCELYPEIC lextual
2135	attribute shall be represe	ant of using the
2130		ented using the
2137	objects by the initial ta	of the jmatchibutevalueAsoclets
2130	objects by the initial tag	j. Integer. or Octets. ,
2139	(modiumCongumed) and go have	attributes use both objects
2140		ve both tags.
2141 2142	If the imattributeWalueAc	Integer object is not used (no
2142 2142	II the juactificatevalueas.	- aball return the value (10
2143 2144	indicating other If the	interviewe (-1)
2144 2145	ugod (no 'Ogtota:' tog)	the agent shall return a gere-length
21+3 21/16	actet string	the agent sharr return a zero-rength
2140 21/7	occet string.	
2147	Thic value is a type 2 en	um "
2140	$\cdots = \int imAttributeEntry 1$	
2177 2150	··- (JMACCIIDUCEENCLY I)	
2150	imattributeInstanceIndex OBJECT-T	vdr
2151		7)
2152	MAX-ACCESS not-accessible	· /
2155	STATUS durrent	
2154		
2155	"A running 16-bit index of	f the attributed of the same type for
2150	A running to-bit fildex 0.	ibutes with only a single instance por
2157	ioh thig index value cha	ll be 1 For those attributes that are
2150	a gingle value per degumos	t the index value shall be the
2137	a singre varue per documen	IC, THE THUCK VALUE SHALL DE THE

[Page 89]

2160 2161 2162 2163 2164 2165 2166	document number, starting with 1 for the first document in the job. Jobs with only a single document shall use the index value of 1. For those attributes that can have multiple values per job and per document, such as documentFormatIndex or documentFormatEnum , the index shall be a running index for the job as a whole, starting at 1.
2167 2168 2169 2170	Each job shall be identified by jmJobIndex value and each job shall be in one job set identified by jmJobSetIndex ." ::= { jmAttributeEntry 2 }
2171	imAttributeValueAsInteger OBJECT-TYPE
2172	SYNTAX $Thteger 32(02147483647)$
2172	MAX-ACCESS read-only
2173	STATUS current
2175	DESCRIPTION
2176	"The integer value of the attribute The value of the attribute
2177	shall be represented as an integer if the enum description
2178	JMAttributeTypeTC definition (see JMAttributeTypeTC on page 37)
2179	has the tag: 'Integer:'
2180	nab ene eage inceger .
2181	Depending on the enum definition, this object value may be an
2182	integer, a counter, an index, or an enum, depending on the
2183	jmAttributeTypeIndex value. The units of this value are
2184	specified in the enum description.
2185	
2186	For those attributes that are accumulating job consumption as
2187	the job is processed as specified in the JmAttributeTypeTC ,
2188	shall contain the final value after the job completes
2189	processing, i.e., this value shall indicate the total usage of
2190	this resource made by the job.
2191	
2192	A monitoring application is able to copy this value to a
2193	suitable longer term storage for later processing as part of an
2194	accounting system.
2195	
2196	Since the agent may add attributes representing resources to
2197	this table while the job is waiting to be processed or being
2198	processed, which can be a long time before any of the resources
2199	in the value of the agent shall set the value of the
2200	hag not wot gongumod
2201	has not yet consumed.
2202	Attributes for which the concept of an integer value is
2203	meaningless, such as fileName interpreter , and
2205	physical DeviceName, do not have the 'Integer:' tag in the
2206	JMAttributeTypeTC definition and so shall return a value of (-1)
2207	to indicate other for jmAttributeValueAsInteger."
2208	::= { jmAttributeEntry 3 }
2209	
2210	jmAttributeValueAsOctets OBJECT-TYPE
2211	SYNTAX OCTET STRING(SIZE(063))
2212	MAX-ACCESS read-only

2213 2214 2215 2216 2217 2218 2219	<pre>STATUS current DESCRIPTION "The octet string value of the attribute. The value of the attribute shall be represented as an OCTET STRING if the enum description JmAttributeTypeTC definition (see JmAttributeTypeTC on page 37) has the tag: 'Octets:'.</pre>
2220 2221 2222 2223	Depending on the enum definition, this object value may be a coded character set string (text) or a binary octet string, such as DateAndTime .
2224 2225 2226 2227 2228 2229 2230 2231	<pre>Attributes for which the concept of an octet string value is meaningless, such as pagesCompleted, do not have the tag 'Octets:' in the JmAttributeTypeTC definition and so shall return a value of a zero length string for jmAttributeValueAsOctets." ::= { jmAttributeEntry 4 }</pre>

```
2232
      -- Conformance Information
2233
2234
      jmMIBConformance OBJECT IDENTIFIER ::= { jobmonmib 2 }
2235
2236
      -- compliance statements
2237
      jmMIBCompliance MODULE-COMPLIANCE
2238
          STATUS current
2239
          DESCRIPTION
2240
              "The compliance statement for agents that implement the
              job monitoring MIB."
2241
2242
          MODULE -- this module
2243
          MANDATORY-GROUPS {
2244
              jmGeneralGroup, <del>jmCompletedGroup,</del> jmJobIDGroup, jmJobStateGroup,
2245
              jmAttributeGroup }
2246
2247
              OBJECT
                       jmJob<mark>Current</mark>State
2248
              SYNTAX
                           INTEGER {
                    processing(57),
                     needsAttention(79),
                     canceled(8)
                     completed(917)
2249
2250
          DESCRIPTION
2251
              "It is conformant for an agent to implement just these fourthree
2252
              states in this object. Any additional states are conditionally
2253
              mandatory, i.e., an agent shall represent any additional states
              that the server or device implementsoptional. However, a client
2254
2255
              shall accept all of the states from an agent."
2256
2257
              -- OBJECT jmAttributeTypeIndex
2258
                             INTEGER {
              -- SYNTAX
2259
              -- jobState(2)
2260
                    numberOfInterveningJobs(5)
              __
2261
                    deviceAlertCode(6)
              --
2262
                  jobKOctetsRequested(30)
              ___
                    jobKOctetsCompleted(31)
2263
              ___
2264
              ___
                    impressionsRequested(35)
2265
                    impressionsCompleted(36)
              ___
2266
                    outputBinName(22)
              --
2267
              ___
2268
          -- DESCRIPTION
2269
              --"It is conformant for an agent to implement just these 8
              -- attributes. Any additional attributes are conditionally
2270
              -- mandatory, i.e., an agent shall represent any additional
2271
              -- states that the server or device implements. However, a
2272
2273
              -- client shall accept all of the attributes from an agent and
2274
              -- either display them to its user or ignore them.
2275
2276
              -- NOTE - SMI does not allow an enum to be declared as mandatory
2277
              -- if that enum is not a member of a group, but
2278
              -- jmAttributeTypeIndex cannot be a member of a group and still
              -- be not-accessible. So comment the mandatory attributes as if
2279
              -- SMI allowed such a declaration in order to declare the
2280
```

2281	<pre> mandatory attributes."</pre>
2282	
2283	There are no conditionally mandatory or optional groups.
2284	
2285	
	the jmQueueGroup is conditionally mandatory. An agent shall
	implement the imOueueGroup if the server or device that the
	- agent instruments performs queuing.
2286	::= { imMIBConformance 1 }
2287	
2288	imMIRGroups OBJECT IDENTIFIER ::= { imMIRConformance 2 }
2280	
2202	imCeneralCroup OBJECT_CPOUD
2200	
2291	UDUECIS {
2292	jmgeneralJobsechame, jmgeneralJobsechame, indenenalNeuNumberOfJobs
2293	JmgeneralAttributePersistence, JmgeneralMaxNumberorJobs,
2294	jmgeneralNumberOIActiveJobs ToComplete ,
2295	jmGeneralOldestActiveJobIndex,
2296	jmGeneralNumberOiJobsCompleted, jmGeneralNewestActiveJobIndex}
2297	STATUS current
2298	DESCRIPTION
2299	"The general group."
2300	::= { jmMIBGroups 1 }
2301	
2302	jmQueueGroup_OBJECT_GROUP
2303	OBJECTS {
2304	jmQueueJobIndex, jmQueueNumberOfInterveningJobs, jmJobPriority,
2305	jmJobProcessAfterDateAndTime }
2306	STATUS current
2307	DESCRIPTION
2308	"The queue group conditionally mandatory."
2309	\div = { $jmMIBGroups 2$ }
2310	
2311	imCompletedGroup OBJECT GROUP
2312	OBTECTS {
2313	imCompletedJobIndex
2314	STATUS ourrent
2315	
2315	"The completed group "
2310	$\therefore = \int \frac{1}{2} M T P C round 2$
2317	
2310	TATATATA
2319	JMJODIDGroup OBJECI-GROUP
2320	
2321	
2322	STATUS current
2323	DESCRIPTION
2324	"The job ID group."
2325	::= { jmMIBGroups 2 }
2326	
2327	jmJob <mark>State</mark> Group OBJECT-GROUP
2328	OBJECTS {
2329	jmJobName, jmJobIdName, jmJobIdNumber, jmJobServiceTypes,
2330	jmJobOwner, jmJobDeviceNameOrQueueRequested, jmJob Current State,

Job Monitoring MIB, V0.871April 4, 1997

2331	jmJobStateKOctetsCompleted, jmJobStateImpressionsCompleted,
2332	jmJobStateAssociatedValue jmJobStateReasons
2333	STAT <mark>US</mark> current
2334	DESCRIPTION
2335	"The job <mark>state</mark> group."
2336	::= { jmMIBGroups 34 }
2337	
2338	jmAttributeGroup OBJECT-GROUP
2339	OBJECTS {
2340	jmAttributeValueAsInteger, jmAttributeValueAsOctets }
2341	STATUS current
2342	DESCRIPTION
2343	"The attribute group."
2344	$::= \{ jmMIBGroups 5 \}$
2345	
2346	
2347	END

Appendix A - Job Life Cycle

2349 **11. Job Life Cycle**

2348

2350	The job object has well-defined states and client operations that affect the transition between the
2351	job states. Internal server and printer actions also affect the transitions of the job between the job
2352	states. These states and transitions are referred to as the job's life cycle.
2353	Not all implementations of job submission protocols have all of the states of the job model
2354	specified here. The job model specified here is intended to be a superset of most implementations.
2355	It is the purpose of the agent to map the particular implementation's job life cycle onto the one
2356	specified here. The agent may omit any states not implemented. Only the processing,
2357	needsAttention, canceled, and completed states are required to be implemented by an agent.
2358	However, a management application shall be prepared to accept any of the states in the job life
2359	cycle specified here, so that the management application can interoperate with any conforming
2360	agent.
2361	The job states are intended to be the user visible. The agent shall make these states visible in the
2362	MIB, but only for the subset of job states that the implementation has. Implementations may need
2363	to have sub-states of these user-visible states. Such implementation is not specified in this model,
2364	is not supported by this Job Monitoring MIB, and will vary from implementation to
2365	implementation.
2366	One of the purposes of the job model is to specify what is invariant from implementation to
2367	implementation as far as the MIB specification and the user is concerned. Therefore, job states
2368	are all intended to last a user-visible length of time in most implementations. However, some jobs
2369	may pass through some states in zero time in some situations and/or in some implementations.
2370	The job model does not specify how accounting and auditing is implemented, except to require
2371	that accounting and auditing logs are separate from the job life cycle and last longer than job
2372	objects. Jobs in the completed state are not logs, since jobs in the completed state are accessible
2373	via job submission and/or job management protocol operations and are removed from these job
2374	tables after a site-settable period of time. Accounting information may be copied incrementally to
2375	the accounting logs as a job processes, may be copied while the job is in the retained state, or
2376	may be copied while the job is in the completed state, depending on implementation. The same is
2377	true for auditing logs.
2378	The jmJobCurrentState object and the jobState attribute both specify the standard job states.
2379	The legal job state transitions are shown in the state transition diagram presented in
-	

2380 <u>Table 11-2.</u>

Table 11-2 - Legal Job State Transition Table

2381

2382

		New State						
		"active" jobs						
	unkno held pendi proce print needsAt cance co				compl			
Old state	2	3	ng 4	ssing 5	11g 6	7	8 8	eted 9
unknown		yes	yes	yes	yes			
held			yes	yes	yes		yes	
pending		yes		yes	yes		yes	
processing		yes			yes	yes	yes	yes
printing		yes				yes	yes	yes
needsAttention		yes		yes	yes		yes	
canceled	yes							
completed	yes							

2383

2384 12. <u>Bibliography</u>

23\$5 [1] The Printer MIB - RFC 1579. Also an Internet-Draft.

2386 13. Author's Addresses

2387	Pon Bergman
2307	Ron Derginan
2388	Dataproducts Corp.
2389	
2390	Phone: 805-578-4421
2391	Fax:
2392	Email: rbergman@dpc.com
2393	
2394	
2395	Tom Hastings
2396	Xerox Corporation, ESAE-231
2397	701 S. Aviation Blvd.
2398	El Segundo, CA 90245
2399	
2400	Phone: 310-333-6413
2401	Fax: 310-333-5514
2402	EMail: hastings@cp10.es.xerox.com
2403	
2404	
2405	Scott A. Isaacson
2406	Novell, Inc.

2407	122 E 1700 S
2408	Provo, UT 84606
2409	
2410	Phone: 801-861-7366
2411	Fax: 801-861-4025
2412	EMail: scott_isaacson@novell.com
2413	
2414	
2415	Harry Lewis
2416	IBM Corporation
2417	P.O. Box 1900
2418	Boulder, CO 80301-9191
2419	
2420	Phone: (303) 924-5337
2421	Fax:
2422	Email: harryl@vnet.ibm.com
2423	
2424	
2425	Send comments to:
2426	JMP Mailing List: jmp@pwg.org
2427	
2428	JMP Mailing List Subscription Information:
2429	jmp-request@pwg.org
2430	

Job Monitoring MIB, V0.<u>871</u> April 4, 1997

2431	Other Participants:
2432	Chuck Adams - Tektronix
2433	Jeff Barnett - IBM
2434	Keith Carter, IBM Corporation
2435	Jeff Copeland - QMS
2436	Andy Davidson - Tektronix
2437	Roger deBry - IBM
2438	Mabry Dozier - QMS
2439	Lee Ferrel - Canon
2440	Steve Gebert - IBM
2441	Robert Herriot - Sun Microsystems Inc.
2442	Shige Kanemitsu - Kyocera
2443	David Kellerman - Northlake Software
2444	Rick Landau - Digital
2445	Harry Lewis - IBM
2446	Pete Loya - HP
2447	Ray Lutz - Cognisys
2448	Jay Martin - Underscore
2449	Mike MacKay, Novell, Inc.
2450	Stan McConnell - Xerox
2451	Carl-Uno Manros, Xerox, Corp.
2452	Pat Nogay - IBM
2453	Bob Pentecost - HP
2454	Rob Rhoads - Intel
2455	David Roach - Unisys
2456	Hiroyuki Sato - Canon
2457	Bob Setterbo - Adobe
2458	Gail Songer, EFI
2459	Mike Timperman - Lexmark
2460	Randy Turner - Sharp
2461	William Wagner - Digital Products
2462	Jim Walker - Dazel
2463	Chris Wellens - Interworking Labs
2464	Rob Whittle - Novell
2465	Don Wright - Lexmark
2466	Lloyd Young - Lexmark
2467	Atsushi Yuki - Kyocera
2468	Peter Zehler, Xerox, Corp.

14. Change History (not to be included in the Internet Draft)

2469

2470 All future changes will be recorded here in *reverse* chronological order by version. 2471 14.1 Changes to version 0.7, dated 3/13/97 to make version 0.71, dated 3/26/97 2472 1. Made the formatting changes necessary to make an Internet Draft. 2473 2. Replaced Figure 1 with a Job State Transition table. 2474 3. Clarified that an agent shall not return an SNMP error for an instrumented object, but 2475 shall return the identifies distinguished value. 2476 4. Removed the IMPORT for **PrtInterpreterLangFamilyTC**, since the MIB doesn't 2477 actually use this enum. In fact no enums used in the Attributes table actually need 2478 their enum TC imported into the Job Monitoring MIB, making the Job Monitoring 2479 MIB more extensible for adding new attributes that have textual conventions. The 2480 MIB now imports very little. Only **DateAndTime**, because it is used in the Queue 2481 table. Even the **TimeStamp** TC which is used in the attribute table, need not be 2482 imported into the Job Monitoring MIB. 2483 5. Explained why there is both a **jmJobState** and a **jmJobStateReasons** object: so that 2484 the reasons can be extended without the monitoring application becoming confused as to what is happening, since the states won't be extended. 2485 2486 6. Clarified that **retained** is an optional state and its relationship to the **completed** state. 2487 Added conformance that only the **processing**, **needsAttention**, and **completed** states 2488 are required for conformance. 2489 7. Changed the name of the **jmAttributeValueAsText** object to 2490 **jmAttributeValueAsOctets**, since the **DateAndTime** type is binary, not text. 2491 Changed the tag in the TC from "Text:" to "Octets". 2492 8. Changed the name of the **mediaConsumed**(33) to **mediumConsumed**(33), since 2493 each entry is singular. 2494 14.2 Changes to version 0.6, dated 1/23/97 to make version 0.7, dated 3/13/97 2495 Changes to version 0.6, dated 1/23/97 to make version 0.7, dated 1/29/97: 2496 1. Added PWG agreed boiler plate Status of this Memo. 2497 2. Updated the Abstract from Ron's comments. 2498 3. Incorporated Ron's re-written Introduction. 2499 4. Explained the job set concept as representing a queue within a printer or a server, if 2500 the printer or server has several or the entire set of jobs, if the printer or server has only one queue. 2501 2502 5. Introduced the terminology of "attribute" instead of resource, since our table 2503 represents more than just resources now, as we agreed to move many non-resource Bergman, Hastings, Isaacson, Lewis [Page 100]

2504 2505		objects into it. Changed the name of the group and table from jmResource to jmAttribute .
2506 2507 2508	6.	Clarified that the JmAttributeTypeTC and jmAttributeTable contains information about the job, such as file name, document name, , as well as resources requested and/or consumed. Re-organized the attributes into groups of similar attributes.
2509 2510 2511	7.	Added more explanation about configuration 1 and 2 and added Configuration 3 as agreed to cover the case of a monitoring application that monitors a server not using SNMP while also monitoring using our MIB the printer(s) that the server controls.
2512 2513	8.	Added more explanation of the security, internationalization, and IANA considerations.
2514 2515	9.	Deleted the Job Set Group, since the monitoring application can find all the job sets via a Get.
2516 2517 2518 2519 2520 2521	10	. Removed the jmResourceUnits object and specified the units in each jmAttributeTypeIndex enum. This makes it clearer what the units are and reduces the variability between agent implementations, thus making monitoring applications easier. Also cleanup the attribute names by adding the data type to the attribute name for those attributes that have more than one type that differs in the units (Index vs. Name , Name vs. Enum , DateAndTime vs. TimeStamp).
2522 2523	11	. Added the TimeStamp data type as an alternative to DateAndTime and doubled the number of attributes that have to do with time.
2524 2525	12	. Deleted the JmQueuingAlgorithmTC and JmResourceUnitsTC textual- conventions.
2526 2527	13	Added other (1) and unknown (2) to the JmJobTypesTC and moved the rest of the bits over.
2528	14	. Added other(1) to the JmJobStateTC.
2529	15	. Added jobPrinting (45) to the JmJobStateReasonsTC to align with IPP.
2530 2531 2532 2533 2534 2535 2536 2537	16	. Move 9 objects from the jmJobTable to the JmAttributeTypeTC and jmAttributeTable, making them attributes: jobAccountName, jobComment, jobSourceChannelIndex, physicalDeviceName, jobKOctetsRequested, jobKOctetsCompleted, jobSubmissionDateAndTime, jobSubmissionTime, jobStartedProcessingDateAndTime, jobStartedProcessingTime, jobCompletedDateAndTime, jobCompletedTime. NOTE that some objects became two attributes as we have two forms of time. Also made the end of each name indicate the data type.
2538 2539	17	. Added Requested , Completed , and CompletedCurrentCopy forms for impressions, sheets, and pages attributes.
2540	18	. Added: other(1), outputBin(9) attributes.
2541	19	. Added "CPU" to processingCPUTime attribute.

2542 20. Added jmGeneralJobSetName so that the user could associate a name with a job set
2543 when the implementation had more than one job set. The name would typically be the
2544 queue name in such a case.

- 2545 21. Added **jmGeneralNumberOfJobsCompleted** and renamed
- jmGeneralCurrentNumberOfJobs to jmGeneralNumberOfJobsToComplete, so that
 a monitoring application can find out how many jobs have completed for the
 imCompletedTable and how many are still to be comppleted. Their sum in the total
- 2549 number of jobs in the **jmJobTable**.
- 2550 22. Clarified that **jmQueueIndex** shall be monitonically increasing which can change as new job arrive or the configuration changes.
- 2552 23. Added the word **Queue** to make **jmQueueJobIndex** in the Queue table.
- 24. Clarifed that the jmQueueJobIndex and jmJobIndex shall not be 0 as required by
 SNMP for indexes. This gives agents that want to use the job-identifier that is
 generated by the system as the value for the jmJobIndex and jmQueueJobIndex a
 problem, if 0 is a legal value, such as in LPD.
- 2557 25. Clarified the distinction betwen jmJobName and jmJobComment (now jobComment attribute): jmJobName is more of a name for identification purposes while jobComment is free form text that often isn't present and is intended to convey anything the submitting user wanted to convey usually to him/herself.
- 2561 26. Clarified that -2 (unknown) shall be returned if the value of jmJobIndexNumber is2562 unknown as in the Printer MIB convention.
- 2563 27. Added "OrQueue" to make jmJobDeviceNameOrQueueRequested, since some
 2564 didn't know which object to use for a system in which the user specifies a queue.
- 2565 28. Added upper bound in **jmJobIndex** so that the MIB would compile.
- 2566 29. Added "Index" to make jmAttributeTypeIndex object, since this object is both a2567 type and an index.
- 2568 30. Changed the name of the jmResourceIndex to jmAttributeInstanceIndex, since this
 2569 index can be used for attributes that can have more than one instance per job, such as
 2570 fileName, documentFormat, outputBin, etc.
- 2571 31. Clarified that the jmAttributeInstanceIndex shall be the document number for those
 2572 attributes that are one to one with a document, such as **fileName**(3) and
 2573 **documentName**(4).
- 2574 32. Replaced the jmResourceAmount with jmAttributeValueAsInteger and
 2575 jmAttributeValueAsText

2576 **15. 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 not special prefix.

		2621	, 83
2581	_ C _	2622	, 81
2501	C	2623	, 81
2582	colorantConsumedIndex, 52	2624	, 84
2583	colorantConsumedName, 52	2625	, 74
2584	colorantRequestedIndex, 52	2626	, 74
2585	colorantRequestedName, 52	2627	, 83
	····· I ······ I ······ ··············	2628	JmJobServiceTypesTC, 54
2596	D	2629	, 69, 79
2386	— D —	2630	jmJobStateAssociatedValue, 86
2587	deviceAlertCode 41	2631	imJobStateImpressionsCompleted. 86
2588	documentConjesPoquested 47	2632	imJobStateKOctetsCompleted. 86
2589	documentCopiesKequesteu, 47	2633	. 85
2500	documentFormatIndex 46	2634	JmJobStateReasonsTC, 56
2590	document Formatindex, 40	2635	JmJobStateTC 32
2391	documentiname, 44	2636	im JobSubmission IDIndex 78
		2637	73
2592	— F —	2638	73
2502		2639	73
2593	fileName, 44	2640	, 75 ImTimeIntervalTC 32
		2641	ImTimeTC 22
2594	_I_	20+1 26/2	jin 1 me IC, 32 job A account Name 12
	_	20+2	jobAccountivanie, 43
2595	impressionsCompleted, 50	2043	jobComment, 44
2596	<pre>impressionsCompletedCurrentCopy, 50</pre>	2044	jobCompletedDateAndTime, 55
2597	impressionsInterpreted, 50	2045	jobCompleted 1 lme, 54
2598	impressionsRequested, 50	2040	jobCopiesCompleted, 4/
2599	impressionsSentToDevice, 50	2047	jobCopiesRequested, 47
2600	impressionsSpooled, 50	2040	jobKOctetsCompleted, 49
		2049	jobKOctetsRequested, 48
2601	I	2030	JobName, 41
2001		2001	jobOwner, 43
2602	jmAttributeInstanceIndex, 89	2652	jobPriority, 45
2603	jmAttributeTypeIndex, 89	2653	jobProcessAfterDateAndTime, 45
2604	JmAttributeTypeTC, 37	2654	jobServiceTypes, 42
2605	imAttributeValueAsInteger, 90	2655	jobSourceChannelIndex, 43
2606	jmAttributeValueAsOctets, 90	2656	jobStartedBeingHeldTime, 53
2607	.75	2657	jobStartedProcessingDateAndTime, 53
2608	. 76	2658	jobStartedProcessingTime, 53
2609	imGeneralAttributePersistence, 69	2659	jobState, 39
2610	imGeneral.IobPersistence, 69	2660	jobStateAssociatedValue, 39
2611	imGeneralIohSetName 69	2661	jobStateReasons, 40
2612	70	2662	jobSubmissionDateAndTime, 53
2613	imGeneralNewestActiveIohIndex 71	2663	jobSubmissionTime, 53
2614	imGeneralNumberOfActiveIobs 70		
2615	70	2661	M
2616	, / 0 imGeneralOldestActiveIohIndex 71	2004	
2617	im Joh Indox 70	2665	mediumConsumed, 51
2618	jingoonuca, 79 im JohStata 8/	2666	mediumRequested, 51
2610	jiii ubstate, 04		▲ / -
2620	82		
2020	, 02		

2668 numberOfInterveningJobs, 41

2670 other, 39 2671 outputBinIndex, 46

2672 outputBinName, 46

- 2674 pagesCompleted, 51
- 2686

- pagesCompletedCurrentCopy, 51
- 2675 2676 pagesRequested, 51
- 2677 physicalDeviceIndex, 44
- 2678 physicalDeviceName, 44
- 2679 processingCPUTime, 54 2680 processingMessage, 41
- 2681
- -**S**---
- 2682 sheetsCompleted, 51
 - sheetsCompletedCurrentCopy, 51
- 2683 2684 sheetsRequested, 51
- 2685 sides, 46