INTERNET-DRAFT R. Bergman
Dataproducts Corp.
T. Hastings
Xerox Corporation
S. Isaacson
Novell, Inc.
H. Lewis
IBM Corp.
February 3, 1998

Job Monitoring MIB - V1

<draft-ietf-printmib-job-monitor-07.txt>

Status of this Memo

Abstract

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

TABLE OF CONTENTS

Job Monitoring MIB

1. Introduction

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

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

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

1.1 Types of Information in the MIB

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

User:

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

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

Provide a timely indication that the job has completed and where it can be 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 direct-connect (serial or parallel) and networked devices without unnecessary overhead or complexity, while also providing for higher end systems and devices.

1.2 Types of Job Monitoring Applications

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

1. Monitor a single job starting when the job is submitted and ending a defined period after the job completes. The Job Submission ID table provides the map to find the specific job to be monitored.
2. Monitor all 'active' jobs in a queue, which this specification generalizes to a "job set". End users may use such a program when selecting a least busy printer, so the MIB is designed for such a program to start up quickly and find the information needed quickly without having to read all (completed) jobs in order to find the active jobs. System operators may also use such a program, in which case it would be running for a long period of time and may also be interested in the jobs that have completed. Finally such a program may be used to provide an enhanced console and logging capability.
3. Collect resource usage for accounting or system utilization purposes 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). Such a program is also expected to keep a shadow copy of the entire Job Attribute table including completed, canceled, and aborted jobs which the program updates on each polling cycle. Such a program polls at the rate of the persistence of the Attribute table. The design is not optimized to help such an application determine which jobs are completed, canceled, or aborted. Instead, the application SHALL query each job that the application's shadow copy shows was not complete, canceled, or aborted at the previous poll cycle to see if it is now complete or canceled, plus any new jobs that have been submitted.

The MIB provides a set of objects that represent a compatible subset of job and document attributes of the ISO DPA standard[iso-dpa] and the Internet Printing Protocol (IPP)[ipp-model], so that coherence is maintained between these two protocols and the information presented to end users and system operators by monitoring applications. 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)[tipsi], as well as with ones that do implement ISO DPA.

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

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

2. Terminology and Job Model

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

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

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

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

Attribute: A name, value-pair that specifies a job or document instruction, a status, or a condition of a job or a document that has been submitted to a server or device. A particular attribute NEED NOT be present in each job instance. In other 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 document data contained an embedded attribute, or (3) the server or device supplied a default value. An agent SHALL represent an attribute as an entry (row) in the Attribute table in this MIB in which entries are present only when necessary. Attributes are identified in this MIB by an enum.

Client: 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 over a serial or parallel port to a directly-connected device or over the network to a networked-connected device.

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

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

Document Instruction: An instruction specifying how to process the document. 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.

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

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

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

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

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

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

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

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

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

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

Job Set: A group 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 embed the SNMP agent in the device, the MIB job set normally represents *all* the jobs known to the device, so that the implementation only implements a single job set. If the 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.

Monitor: Short for Job Monitoring Application.

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

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

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

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

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

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

SNMP Information Object: 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.

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

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

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

Supervisor: 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.

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

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

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

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

2.1 System Configurations for the Job Monitoring MIB

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

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

2.1.1 Configuration 1 - client-printer

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

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

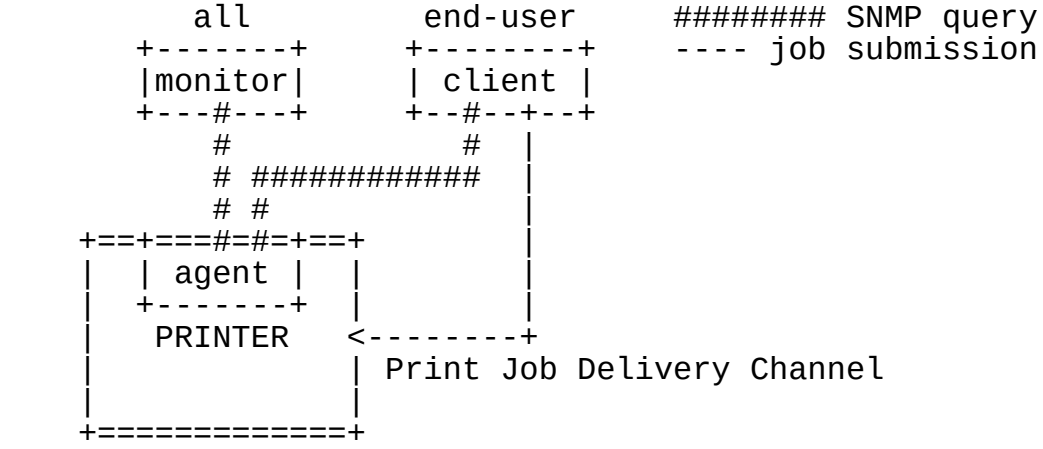

Figure 2-1 - Configuration 1 - client-printer - agent in the printer

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

1. Multiple clients MAY submit jobs to a printer.
2. Multiple clients MAY monitor a printer.
3. Multiple monitors MAY monitor a printer.
4. A client MAY submit jobs to multiple printers.
5. A monitor MAY monitor multiple printers.

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

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

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

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

There is no SNMP Job Monitoring MIB agent in the printer in configuration 2, at least that the client or monitor are aware. In this configuration, the agent SHALL return the 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. The Job Monitoring MIB agent SHALL obtain the required information from the printer by a method that is beyond the scope of this document. The agent in the server SHALL keep the job in the Job Monitoring MIB in the server as long as the job is in the printer, plus a defined time period after the job enters the completed state in which accounting programs can copy out the accounting data from the Job Monitoring MIB.

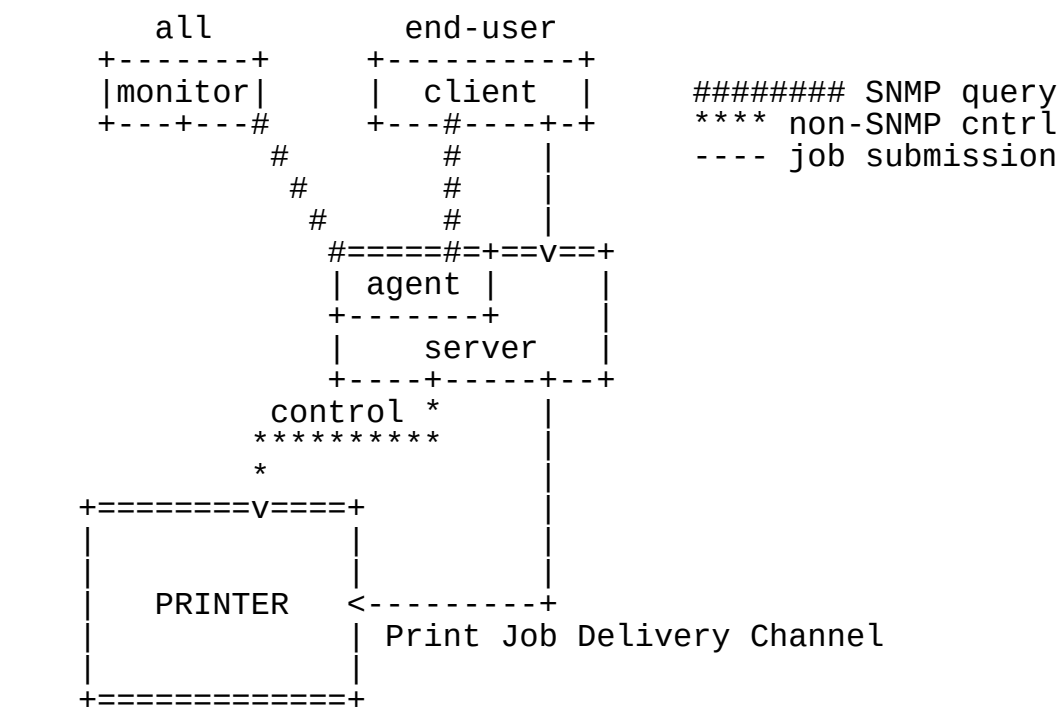

Figure 2-2 - Configuration 2 - client-server-printer - agent in the server

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

1. Multiple clients MAY submit jobs to a server.
2. Multiple clients MAY monitor a server.
3. Multiple monitors MAY monitor a server.
4. A client MAY submit jobs to multiple servers.
5. A monitor MAY monitor multiple servers.
6. Multiple servers MAY submit jobs to a printer.
7. Multiple servers MAY control a printer.

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

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

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

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

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

In configuration 3, the agent (in the printer) SHALL keep the values of the objects in the Job Monitoring MIB that the agent implements updated for a job that the server has submitted to the printer. The agent SHALL obtain information about the jobs submitted to the printer from the server (either in the job submission protocol, in 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 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 out the accounting data from the Job Monitoring MIB.

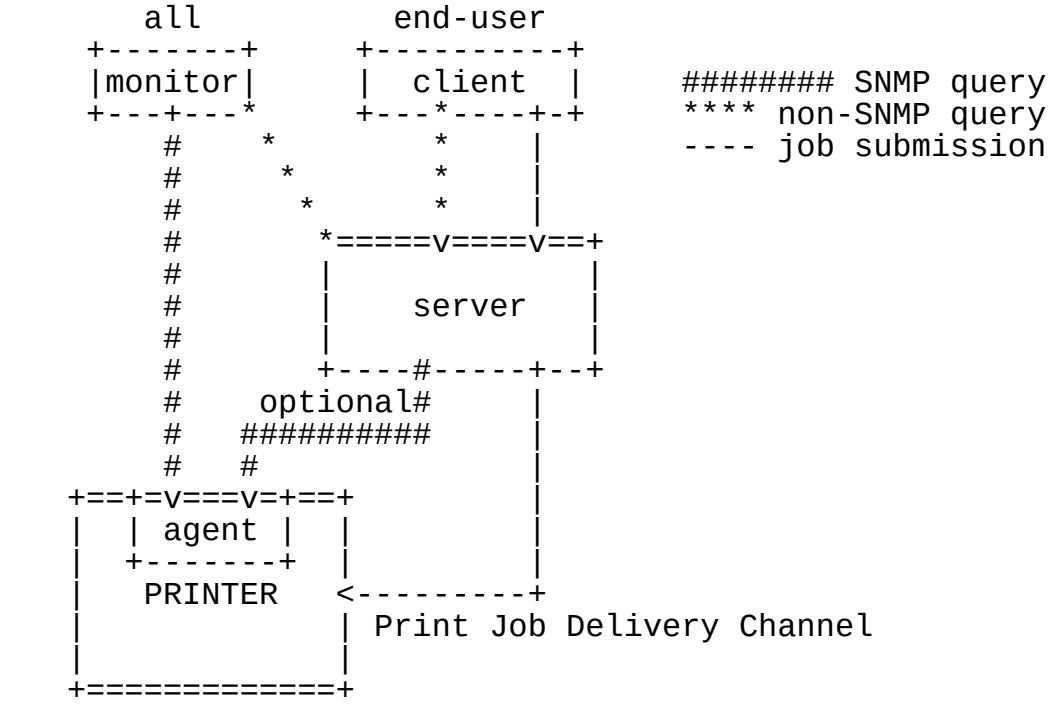

Figure 2-3 - Configuration 3 - client-server-printer - client monitors printer agent and server

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

1. Multiple clients MAY submit jobs to a server.
2. Multiple clients MAY monitor a server.
3. Multiple monitors MAY monitor a server.
4. A client MAY submit jobs to multiple servers.
5. A monitor MAY monitor multiple servers.
6. Multiple servers MAY submit jobs to a printer.
7. Multiple servers MAY control a printer.

3. Managed Object Usage

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

3.1 Conformance Considerations

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

3.1.1 Conformance Terminology

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

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

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

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

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

3.1.2 Agent Conformance Requirements

A conforming agent:

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

NOTE - This MIB, like the Printer MIB, is written following the subset of SMIv2 that can be supported by SMIv1 and SNMPv1 implementations.

3.1.2.1 MIB II System Group objects

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

3.1.2.2 MIB II Interface Group objects

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

3.1.2.3 Printer MIB objects

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

3.1.3 Job Monitoring Application Conformance Requirements

A conforming job monitoring application:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

3.3 The Attribute Mechanism

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

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

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

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

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

3.3.1 Conformance of Attribute Implementation

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

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

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

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

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

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

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

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

3.3.3 Data Sub-types and Attribute Naming Conventions

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

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

'Int32(-2..)'	Integer32 (-2..2147483647)
'Int32(0..)'	Integer32 (0..2147483647)
'Int32(1..)'	Integer32 (1..2147483647)
'Int32(m..n)'	For all other Integer ranges, the lower and upper bound of the range is indicated.
'UTF8String63'	JmUTF8StringTC (SIZE(0..63))
'JobString63'	JmJobStringTC (SIZE(0..63))
'Octets63'	OCTET STRING (SIZE(0..63))
'Octets(m..n)'	For all other OCTET STRING ranges, the exact range is indicated.

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

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

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

3.3.5 Requested Objects and Attributes

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

3.3.6 Consumption Attributes

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

3.3.7 Index Value Attributes

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

3.4 Monitoring Job Progress

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

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

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

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

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

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

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

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

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

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

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

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

Job Collation Type = uncollatedSheets(3)

jmJobImpressionsCompleted	ImpressionsCompletedCurrentCopy	sheetCompletedCopyNumber	sheetCompletedDocumentNumber
0	0	0	0
1	1	1	1
2	1	2	1
3	1	3	1
4	2	1	1
5	2	2	1
6	2	3	1
7	3	1	1
8	3	2	1
9	3	3	1
10	1	1	2
11	1	2	2
12	1	3	2
13	2	1	2
14	2	2	2
15	2	3	2
16	3	1	2
17	3	2	2
18	3	3	2

Job Collation Type = collatedDocuments(4)

JmJobImpressionsCompleted	ImpressionsCompletedCurrentCopy	sheetCompletedCopyNumber	sheetCompletedDocumentNumber
0	0	0	0
1	1	1	1
2	2	1	1
3	3	1	1
4	1	1	2
5	2	1	2
6	3	1	2
7	1	2	1
8	2	2	1
9	3	2	1
10	1	2	2
11	2	2	2
12	3	2	2
13	1	3	1
14	2	3	1
15	3	3	1
16	1	3	2
17	2	3	2
18	3	3	2

Job Collation Type = uncollatedDocuments(5)

jmJobImpressionsCompleted	ImpressionsCompletedCurrentCopy	sheetCompletedCopyNumber	sheetCompletedDocumentNumber
0	0	0	0
1	1	1	1
2	2	1	1
3	3	1	1
4	1	2	1
5	2	2	1
6	3	2	1
7	1	3	1
8	2	3	1
9	3	3	1
10	1	1	2
11	2	1	2
12	3	1	2
13	1	2	2
14	2	2	2
15	3	2	2
16	1	3	2
17	2	3	2
18	3	3	2

3.5 Job Identification

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

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

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

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

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

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

3.6 Internationalization Considerations

This section describes the internationalization considerations included in this MIB.

3.6.1 Text generated by the server or device

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

1. jmGeneralJobSetName object
2. processingMessage(6) attribute
3. physicalDevice(32) (name value) attribute

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

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

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

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

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

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

3.6.2 Text supplied by the job submitter

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

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

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

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

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

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

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

3.6.3 'DateAndTime' for representing the date and time

This MIB also contains objects that are represented using the DateAndTime textual convention from SMIv2 [SMIv2-TC]. The job management application SHALL display such objects in the locale of the user running the monitoring application.

3.7 IANA and PWG Registration Considerations

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

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

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

3.7.1 PWG Registration of enums

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

The PWG has defined several type of enumerations for use in the Job Monitoring MIB and the Printer MIB[print-mib]. These types 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:

3.7.1.1 Type 1 enumerations

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

There are no type 1 enums in the current draft.

3.7.1.2 Type 2 enumerations

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

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

1. JmUTF8StringTC
2. JmJobStringTC
3. JmNaturalLanguageTagTC
4. JmTimeStampTC
5. JmFinishingTC [same enum values as IPP "finishing" attribute]
6. JmPrintQualityTC [same enum values as IPP "print-quality" attribute]
7. JmTonerEconomyTC
8. JmMediumTypeTC
9. JmJobSubmissionIDTypeTC
10. JmJobCollationTypeTC
11. JmJobStateTC [same enum values as IPP "job-state" attribute]
12. JmAttributeTypeTC

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

3.7.1.3 Type 3 enumeration

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

There are no type 3 enums in the current draft.

3.7.2 PWG Registration of type 2 bit values

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

1. JmJobServiceTypesTC
2. JmJobStateReasons1TC
3. JmJobStateReasons2TC
4. JmJobStateReasons3TC
5. JmJobStateReasons4TC

These textual-conventions are defined as bits in an Integer so that they can be used with SNMPv1 SMI. The jobStateReasons*N* (*N*=1..4) attributes are defined as bit values using the corresponding JmJobStateReasons*N*TC textual-conventions.

The registration of JmJobServiceTypesTC and JmJobStateReasons*N*TC bit values SHALL follow the procedures for a type 2 enum as specified in Section 3.7.1.2.

3.7.3 PWG Registration of Job Submission Id Formats

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

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

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

3.8 Security Considerations

3.8.1 Read-Write objects

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

3.8.2 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 jmJobKOctetsPerCopyRequested and jmJobKOctetsProcessed objects, so that a user can tell how busy a printer is. Other sites MAY allow all unprivileged users to see all objects of all jobs. This MIB does not require, nor does it specify how, such restrictions would be implemented. A monitoring application SHOULD enforce the site security policy with respect to returning information to an unprivileged end user that is using the monitoring application to monitor jobs that do not belong to that user, i.e., the jmJobOwner object in the jmJobTable does not match the user's user name.

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

3.9 Notifications

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

4. MIB specification

The following pages constitute the actual Job Monitoring MIB.

```
1271  Job-Monitoring-MIB DEFINITIONS ::= BEGIN
1272
1273  IMPORTS
          MODULE-IDENTITY, OBJECT-TYPE, enterprises,
          Integer32                                   FROM SNMPv2-SMI
          TEXTUAL-CONVENTION                          FROM SNMPv2-TC
          MODULE-COMPLIANCE, OBJECT-GROUP             FROM SNMPv2-CONF;
          -- The following textual-conventions are needed to implement
          -- certain attributes, but are not needed to compile this MIB.
          -- They are provided here for convenience:
          -- hrDeviceIndex                       FROM HOST-RESOURCES-MIB
          -- DateAndTime                         FROM SNMPv2-TC
          -- PrtInterpreterLangFamilyTC,
          -- CodedCharSet                        FROM Printer-MIB
1274
1275  -- Use the enterprises arc assigned to the PWG which is pwg(2699).
1276  -- Group all PWG mibs under mibs(1).
1277
1278  jobmonMIB MODULE-IDENTITY
1279      LAST-UPDATED "9802030000Z"
1280      ORGANIZATION "Printer Working Group (PWG)"
1281      CONTACT-INFO
1282          "Tom Hastings
1283          Postal:  Xerox Corp.
1284                   Mail stop ESAE-231
1285                   701 S. Aviation Blvd.
1286                   El Segundo, CA 90245
1287
1288          Tel:     (301)333-6413
1289          Fax:     (301)333-5514
1290          E-mail:  hastings@cp10.es.xerox.com
1291
1292          Send questions and comments to the Printer Working Group (PWG)
1293          using the Job Monitoring Project (JMP) Mailing List:
1294          jmp@pwg.org
1295
1296          For further information, including how to subscribe to the
1297          jmp mailing list, access the PWG web page under 'JMP':
1298
1299              http://www.pwg.org/
1300
1301          Implementers of this specification are encouraged to join the
1302          jmp mailing list in order to participate in discussions on any
1303          clarifications needed and registration proposals being reviewed
1304          in order to achieve consensus."
1305      DESCRIPTION
1306          "The MIB module for monitoring job in servers, printers, and
1307          other devices.
1308
1309          Version: 1.0"
1310      ::= { enterprises pwg(2699)  mibs(1)  jobmonMIB(1) }
```

```
1311
1312  -- Textual conventions for this MIB module
1313
1314  JmUTF8StringTC ::= TEXTUAL-CONVENTION
1315      DISPLAY-HINT "255a"
1316      STATUS      current
1317      DESCRIPTION
1318          "To facilitate internationalization, this TC represents
1319          information taken from the ISO/IEC IS 10646-1 character set,
1320          encoded as an octet string using the UTF-8 character encoding
1321          scheme."
1322      REFERENCE
1323          "See section 3.6.1, entitled: 'Text generated by the server or
1324          device'."
1325      SYNTAX      OCTET STRING (SIZE (0..63))
1326
1327
1328
1329
1330  JmJobStringTC ::= TEXTUAL-CONVENTION
1331      STATUS      current
1332      DESCRIPTION
1333          "To facilitate internationalization, this TC represents
1334          information using any coded character set registered by IANA as
1335          specified in section 3.7.  While it is recommended that the
1336          coded character set be UTF-8 [UTF-8], the actual coded
1337          character set SHALL be indicated by the value of the
1338          jobCodedCharSet(8) attribute for the job."
1339      REFERENCE
1340          "See section 3.6.2, entitled: 'Text supplied by the job
1341          submitter'."
1342      SYNTAX      OCTET STRING (SIZE (0..63))
1343
1344
1345
1346
1347  JmNaturalLanguageTagTC  ::= TEXTUAL-CONVENTION
1348      STATUS      current
1349      DESCRIPTION
1350          "An IETF RFC 1766-compliant 'language tag', with zero or more
1351          sub-tags that identify a natural language.  While RFC 1766
1352          specifies that the US-ASCII values are case-insensitive, this
1353          MIB specification requires that all characters SHALL be lower
1354          case in order to simplify comparing by management
1355          applications."
1356      REFERENCE
1357          "See section 3.6.1, entitled: 'Text generated by the server or
1358          device' and section 3.6.2, entitled: 'Text supplied by the job
1359          submitter'."
1360      SYNTAX      OCTET STRING (SIZE (0..63))
1361
1362
```

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

```
1399
1400  JmFinishingTC ::= TEXTUAL-CONVENTION
1401      STATUS      current
1402      DESCRIPTION
1403          "The type of finishing operation.
1404
1405          These values are the same as the enum values of the IPP
1406          'finishings' attribute.  See Section 3.7.1.2.
1407
1408          other(1),
1409              Some other finishing operation besides one of the specified
1410              or registered values.
1411
1412          unknown(2),
1413              The finishing is unknown.
1414
1415          none(3),
1416              Perform no finishing.
1417
1418          staple(4),
1419              Bind the document(s) with one or more staples. The exact
1420              number and placement of the staples is site-defined.
1421
1422          punch(5),
1423              This value indicates that holes are required in the
1424              finished document. The exact number and placement of the
1425              holes is site-defined  The punch specification MAY be
1426              satisfied (in a site- and implementation-specific manner)
1427              either by drilling/punching, or by substituting pre-drilled
1428              media.
1429
1430          cover(6),
1431              This value is specified when it is desired to select a non-
1432              printed (or pre-printed) cover for the document. This does
1433              not supplant the specification of a printed cover (on cover
1434              stock medium) by the document itself.
1435
1436          bind(7)
1437              This value indicates that a binding is to be applied to the
1438              document; the type and placement of the binding is product-
1439              specific."
1440      REFERENCE
1441          "This is a type 2 enumeration.  See Section 3.7.1.2."
1442      SYNTAX      INTEGER {
1443          other(1),
1444          unknown(2),
1445          none(3),
1446          staple(4),
1447          punch(5),
1448          cover(6),
1449          bind(7)
1450      }
```

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

```
1490
1491   JmTonerEconomyTC ::= TEXTUAL-CONVENTION
1492       STATUS      current
1493       DESCRIPTION
1494           "Toner economy settings."
1495       REFERENCE
1496           "This is a type 2 enumeration.  See Section 3.7.1.2."
1497       SYNTAX      INTEGER {
                        unknown(2),    --  unknown.
                        off(3),        --  Off.  Normal.  Use full toner.
                        on(4)          --  On.  Use less toner than normal.
1498       }
1499
1500
1501
1502   JmBooleanTC ::= TEXTUAL-CONVENTION
1503       STATUS      current
1504       DESCRIPTION
1505           "Boolean true or false value."
1506       REFERENCE
1507           "This is a type 2 enumeration.  See Section 3.7.1.2."
1508       SYNTAX      INTEGER {
                        unknown(2),    --  unknown.
                        false(3),      --  FALSE.
                        true(4)        --  TRUE.
1509       }
1510
1511
1512
1513   JmMediumTypeTC ::= TEXTUAL-CONVENTION
1514       STATUS      current
1515       DESCRIPTION
1516           "Identifies the type of medium.
1517
1518           other(1),
1519               The type is neither one of the values listed in this
1520               specification nor a registered value.
1521
1522           unknown(2),
1523               The type is not known.
1524
1525           stationery(3),
1526               Separately cut sheets of an opaque material.
1527
1528           transparency(4),
1529               Separately cut sheets of a transparent material.
1530
1531           envelope(5),
1532               Envelopes that can be used for conventional mailing
1533               purposes.
```

```
1534
1535            envelopePlain(6),
1536                Envelopes that are not preprinted and have no windows.
1537
1538            envelopeWindow(7),
1539                Envelopes that have windows for addressing purposes.
1540
1541            continuousLong(8),
1542                Continuously connected sheets of an opaque material
1543                connected along the long edge.
1544
1545            continuousShort(9),
1546                Continuously connected sheets of an opaque material
1547                connected along the short edge.
1548
1549            tabStock(10),
1550                Media with tabs.
1551
1552            multiPartForm(11),
1553                Form medium composed of multiple layers not pre-attached to
1554                one another;  each sheet MAY be drawn separately from an
1555                input source.
1556
1557            labels(12),
1558                Label-stock.
1559
1560            multiLayer(13)
1561                Form medium composed of multiple layers which are pre-
1562                attached to one another, e.g. for use with impact
1563                printers."
1564        REFERENCE
1565            "This is a type 2 enumeration.  See Section 3.7.1.2.  These
1566            enum values correspond to the keyword name strings of the
1567            prtInputMediaType object in the Printer MIB [print-mib].  There
1568            is no printer description attribute in IPP/1.0 that represents
1569            these values."
1570        SYNTAX      INTEGER {
1571            other(1),
1572            unknown(2),
1573            stationery(3),
1574            transparency(4),
1575            envelope(5),
1576            envelopePlain(6),
1577            envelopeWindow(7),
1578            continuousLong(8),
1579            continuousShort(9),
1580            tabStock(10),
1581            multiPartForm(11),
1582            labels(12),
1583            multiLayer(13)
1584        }
1585
```

```
1586
1587  JmJobCollationTypeTC ::= TEXTUAL-CONVENTION
1588      STATUS      current
1589      DESCRIPTION
1590          "This value is the type of job collation.  Implementations that
1591          don't support multiple documents or don't support multiple
1592          copies SHALL NOT support the uncollatedDocuments(5) value."
1593      REFERENCE
1594          "This is a type 2 enumeration.  See Section 3.7.1.2. See also
1595          Section 3.4, entitled 'Monitoring Job Progress'."
1596      SYNTAX      INTEGER {
1597          other(1),
1598          unknown(2),
1599          uncollatedSheets(3),    -- sheets within each document copy
1600                                  -- are not collated: 1 1 ..., 2 2 ...,
1601          collatedDocuments(4),   -- internal collated sheets,
1602                                  -- documents: A, B, A, B, ...
1603          uncollatedDocuments(5)  -- internal collated sheets,
1604                                  -- documents: A, A, ..., B, B, ...
1605      }
1606
1607
1608  JmJobSubmissionIDTypeTC ::= TEXTUAL-CONVENTION
1609      STATUS      current
1610      DESCRIPTION
1611          "Identifies the format type of a job submission ID.
1612
1613          Each job submission ID is a fixed-length, 48-octet printable
1614          US-ASCII [US-ASCII] coded character string containing no
1615          control characters, consisting of the following fields:
1616
1617            octet  1:  The format letter identifying the format.  The US-
1618              ASCII characters '0-9', 'A-Z', and 'a-z' are assigned in
1619              order giving 62 possible formats.
1620            octets 2-40:  A 39-character, US-ASCII trailing SPACE filled
1621              field specified by the format letter, if the data is less
1622              than 39 ASCII characters.
1623            octets 41-48:  A sequential or random US-ASCII number to make
1624              the ID quasi-unique.
1625
1626          If the client does not supply a job submission ID in the job
1627          submission protocol, then the agent SHALL assign a job
1628          submission ID using any of the standard formats that are
1629          reserved for the agent.  Clients SHALL not use formats that are
1630          reserved for agents and agents SHALL NOT use formats that are
1631          reserved for clients, in order to reduce conflicts in ID
1632          generation.  See the description for which formats are reserved
1633          for clients or for agents.
1634
1635          Registration of additional formats may be done following the
1636          procedures described in Section 3.7.3.
1637
```

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

Format Letter	Description
’0’	Job Owner generated by the server/device octets 2-40: The last 39 bytes of the jmJobOwner object. octets 41-48: The US-ASCII 8-decimal-digit sequential number assigned by the agent. This format is reserved for agents.
	NOTE - Clients wishing to use a job submission ID that incorporates the job owner, SHALL use format ’8’, not format ’0’.
’1’	Job Name octets 2-40: The last 39 bytes of the jobName attribute. octets 41-48: The US-ASCII 8-decimal-digit random number assigned by the client. This format is reserved for clients.
’2’	Client MAC address octets 2-40: The client MAC address: in hexadecimal with each nibble of the 6 octet address being ’0’-’9’ or ’A’ - ’F’ (uppercase only). Most significant octet first. octets 41-48: The US-ASCII 8-decimal-digit sequential number assigned by the client. This format is reserved for clients.
’3’	Client URL octets 2-40: The last 39 bytes of the client URL [URI-spec]. octets 41-48: The US-ASCII 8-decimal-digit sequential number assigned by the client. This format is reserved for clients.
’4’	Job URI octets 2-40: The last 39 bytes of the URI [URI-spec] assigned by the server or device to the job when the job was submitted for processing. octets 41-48: The US-ASCII 8-decimal-digit sequential number assigned by the agent. This format is reserved for agents.
’5’	POSIX User Number octets 2-40: The last 39 bytes of a user number, such as POSIX user number. octets 41-48: The US-ASCII 8-decimal-digit sequential number assigned by the client. This format is reserved for clients.

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

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

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

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

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

'B' NetWare PServer
octets 2-40: Contains the Directory Path Name as recorded by the Novell File Server in the queue directory. If the string is less than 40 octets, the left-most character in the string shall appear in octet position 2. Otherwise, only the last 39 bytes shall be included. Any unused portion of this field shall be filled with spaces.
octets 41-48: '000XXXXX' The US-ASCII representation of the Job Number as per the NetWare File Server Queue Management Services.
This format is reserved for agents.

'C' Server Message Block protocol (SMB)
octets 2-40: Contains a decimal (US-ASCII coded) representation of the 16 bit SMB Tree Id field, which uniquely identifies the connection that submitted the job to the printer. The most significant digit of the numeric string shall be placed in octet position 2. All unused portions of this field shall be filled with spaces. The SMB Tree Id has a maximum value of 65,535.
octets 41-48: The US-ASCII 8-decimal-digit leading zero representation of the File Handle returned from the device to the client in response to a Create Print File command.
This format is reserved for agents.

'D' Transport Independent Printer/System Interface (TIP/SI)
octets 2-40: Contains the Job Name from the Job Control-Start Job (JC-SJ) command. If the Job Name portion is less than 40 octets, the left-most character in the string shall appear in octet position 2. Any unused portion of this field shall be filled with spaces. Otherwise, only the last 39 bytes shall be included.
octets 41-48: The US-ASCII 8-decimal-digit leading zero representation of the jmJobIndex assigned by the agent.
This format is reserved for agents, since the agent supplies octets 41-48, though the client supplies the job name. See format '1' reserved to clients to submit job name ids in which they supply octets 41-48.

'E' IPDS on the MVS or VSE platform

octets 2-40: Contains bytes 2-27 of the XOH Define Group Boundary Group ID triplet. Octet position 2 MUST carry the value x'01'. Bytes 28-40 MUST be filled with spaces.
octets 41-48: The US-ASCII 8-decimal-digit leading zero representation of the jmJobIndex assigned by the agent.
This format is reserved for agents, since the agent supplies octets 41-48, though the client supplies the job name.

'F' IPDS on the VM platform
octets 2-40: Contains bytes 2-31 of the XOH Define Group Boundary Group ID triplet. Octet position 2 MUST carry the value x'02'. Bytes 32-40 MUST be filled with spaces.
octets 41-48: The US-ASCII 8-decimal-digit leading zero representation of the jmJobIndex assigned by the agent.
This format is reserved for agents, since the agent supplies octets 41-48, though the client supplies the file name.

```
1788          'G' IPDS on the OS/400 platform
1789          octets 2-40:  Contains bytes 2-36 of the XOH Define Group
1790              Boundary Group ID triplet.  Octet position 2 MUST carry the
1791              value x'03'.  Bytes 37-40 MUST be filled with spaces.
1792          octets 41-48: The US-ASCII 8-decimal-digit leading zero
1793              representation of the jmJobIndex assigned by the agent.
1794          This format is reserved for agents, since the agent supplies
1795              octets 41-48, though the client supplies the job name.
1796
1797          NOTE - the job submission id is only intended to be unique
1798          between a limited set of clients for a limited duration of
1799          time, namely, for the life time of the job in the context of
1800          the server or device that is processing the job.  Some of the
1801          formats include something that is unique per client and a
1802          random number so that the same job submitted by the same client
1803          will have a different job submission id.  For other formats,
1804          where part of the id is guaranteed to be unique for each
1805          client, such as the MAC address or URL, a sequential number
1806          SHOULD suffice for each client (and may be easier for each
1807          client to manage).  Therefore, the length of the job submission
1808          id has been selected to reduce the probability of collision to
1809          an extremely low number, but is not intended to be an absolute
1810          guarantee of uniqueness.  None-the-less, collisions are
1811          remotely possible, but without bad consequences, since this MIB
1812          is intended to be used only for monitoring jobs, not for
1813          controlling and managing them."
1814      REFERENCE
1815          "This is like a type 2 enumeration.  See section 3.7.3."
1816      SYNTAX    OCTET STRING(SIZE(1)) -- ASCII '0'-'9', 'A'-'Z', 'a'-'z'
```

```
JmJobStateTC ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION
        "The current state of the job (pending, processing, completed,
        etc.).

        The following figure shows the normal job state transitions:

                                                          +----> canceled(7)
                                                         /
    +---> pending(3) -------> processing(5) ------+------> completed(9)
    |         ^                    ^              \
--->+         |                    |               +----> aborted(8)
    |         v                    v              /
    +---> pendingHeld(4)  processingStopped(6) ---+
```

Figure 4 - Normal Job State Transitions

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

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

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

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

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

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

JmJobStateReasons*N*TC (*N*=1..4) textual convention for the specification of each reason.

processing(5),
One or more of:

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

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

OR

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

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

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

processingStopped(6),
The job has stopped while processing for any number of reasons and will return to the processing state as soon as the reasons are no longer present.

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

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

```
1920                device MIB using the job's deviceIndex attribute(s), if the
1921                agent implements such a device MIB
1922
1923            canceled(7),
1924                A client has canceled the job and the server or device has
1925                completed canceling the job AND all MIB objects and
1926                attributes have reached their final values for the job.
1927                While the server or device is canceling the job, the job's
1928                jmJobStateReasons1 object SHOULD contain the
1929                processingToStopPoint value and one of the canceledByUser,
1930                canceledByOperator, or canceledAtDevice values.  The
1931                canceledByUser, canceledByOperator, or canceledAtDevice
1932                values remain while the job is in the canceled state.
1933
1934            aborted(8),
1935                The job has been aborted by the system, usually while the
1936                job was in the processing or processingStopped state and
1937                the server or device has completed aborting the job AND all
1938                MIB objects and attributes have reached their final values
1939                for the job.  While the server or device is aborting the
1940                job, the job's jmJobStateReasons1 object MAY contain the
1941                processingToStopPoint and abortedBySystem values.  If
1942                implemented, the abortedBySystem value SHALL remain while
1943                the job is in the aborted state.
1944
1945            completed(9)
1946                The job has completed successfully or with warnings or
1947                errors after processing and all of the media have been
1948                successfully stacked in the appropriate output bin(s) AND
1949                all MIB objects and attributes have reached their final
1950                values for the job.  The job's jmJobStateReasons1 object
1951                SHOULD contain one of: completedSuccessfully,
1952                completedWithWarnings, or completedWithErrors values."
1953        REFERENCE
1954            "This is a type 2 enumeration.  See Section 3.7.1.2."
1955        SYNTAX      INTEGER {
1956            unknown(2),
1957            pending(3),
1958            pendingHeld(4),
1959            processing(5),
1960            processingStopped(6),
1961            canceled(7),
1962            aborted(8),
1963            completed(9)
1964        }
```

```
1965
1966   JmAttributeTypeTC ::= TEXTUAL-CONVENTION
1967       STATUS      current
1968       DESCRIPTION
1969           "The type of the attribute which identifies the attribute.
1970
1971           In the following definitions of the enums, each description
1972           indicates whether the useful value of the attribute SHALL be
1973           represented using the jmAttributeValueAsInteger or the
1974           jmAttributeValueAsOctets objects by the initial tag: 'INTEGER:'
1975           or 'OCTETS:', respectively.
1976
1977           Some attributes allow the agent implementer a choice of useful
1978           values of either an integer, an octets representation, or both,
1979           depending on implementation.  These attributes are indicated
1980           with 'INTEGER:' AND/OR 'OCTETS:' tags.
1981
1982           A very few attributes require both objects at the same time to
1983           represent a pair of useful values (see mediumConsumed(171)).
1984           These attributes are indicated with 'INTEGER:' AND 'OCTETS:'
1985           tags.  See the jmAttributeGroup for the descriptions of these
1986           two MANDATORY objects.
1987
1988           NOTE - The enum assignments are grouped logically with values
1989           assigned in groups of 20, so that additional values may be
1990           registered in the future and assigned a value that is part of
1991           their logical grouping.
1992
1993           Values in the range 2**30 to 2**31-1 are reserved for private
1994           or experimental usage.  This range corresponds to the same
1995           range reserved in IPP.  Implementers are warned that use of
1996           such values may conflict with other implementations.
1997           Implementers are encouraged to request registration of enum
1998           values following the procedures in Section 3.7.1.
1999
2000           NOTE: No attribute name exceeds 31 characters.
2001
2002           The standard attribute types defined at the time of completion
2003           of the specification are:
2004
2005           jmAttributeTypeIndex              Datatype
2006           --------------------              --------
2007
2008           other(1),                         Integer32 (-2..2147483647)
2009                                             AND/OR
2010                                             OCTET STRING(SIZE(0..63))
2011               INTEGER:  and/or  OCTETS:  An attribute that is not in the
2012               list and/or that has not been approved and registered with
2013               the PWG.
```

```
        ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
        + Job State attributes
        +
        + The following attributes specify the state of a job.
        ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

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

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

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

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

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

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

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

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

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

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

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

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

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

```
2108          jobNaturalLanguageTag(9),          OCTET STRING(SIZE(0..63))
2109              OCTETS: The natural language of the job attributes supplied
2110              by the job submitter or defaulted by the server or device
2111              for the job, i.e., all objects and attributes represented
2112              by the 'JmJobStringTC' textual-convention, such as jobName,
2113              mediumRequested, etc.  See Section 3.6.2, entitled 'Text
2114              supplied by the job submitter'.
2115
2116              If the agent does not know what natural language was used
2117              by the job submitting client, the agent SHALL either (1)
2118              return a zero length string value for the
2119              jobNaturalLanguageTag(9) attribute or (2) not return
2120              jobNaturalLanguageTag(9)  attribute for the job.
2121
2122
2123          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2124          + Job Identification attributes
2125          +
2126          + The following attributes help an end user, a system
2127          + operator, or an accounting program identify a job.
2128          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2129
2130          jobURI(20),                        OCTET STRING(SIZE(0..63))
2131              OCTETS:  MULTI-ROW:  The job's Universal Resource
2132              Identifier (URI) [RFC-1738].  See IPP [ipp-model] for
2133              example usage.
2134
2135              NOTE - The agent may be able to generate this value on each
2136              SNMP Get operation from smaller values, rather than having
2137              to store the entire URI.
2138
2139              If the URI exceeds 63 octets, the agent SHALL use multiple
2140              values, with the next 63 octets coming in the second value,
2141              etc.
2142
2143              NOTE - IPP [ipp-model] has a 1023-octet maximum length for
2144              a URI, though the URI standard itself and HTTP/1.1 specify
2145              no maximum length.
2146
2147          jobAccountName(21),                OCTET STRING(SIZE(0..63))
2148              OCTETS:  Arbitrary binary information which MAY be coded
2149              character set data or encrypted data supplied by the
2150              submitting user for use by accounting services to allocate
2151              or categorize charges for services provided, such as a
2152              customer account name or number.
2153
2154              NOTE: This attribute NEED NOT be printable characters.
2155
```

serverAssignedJobName(22), JmJobStringTC (SIZE(0..63))
OCTETS: Configuration 3 only: The human readable string name, number, or ID of the job as assigned by the server that submitted the job to the device that the agent is providing access to with this MIB.

NOTE - This attribute is intended for enabling a user to find his/her job that a server submitted to a device when either the client does not support the jmJobSubmissionID or the server does not pass the jmJobSubmissionID through to the device.

jobName(23), JmJobStringTC (SIZE(0..63))
OCTETS: The human readable string name of the job as assigned by the submitting user to help the user distinguish between his/her various jobs. This name does not need to be unique.

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.

In order to assist users to find their jobs for job submission protocols that don't supply a jmJobSubmissionID, the agent SHOULD maintain the jobName attribute for the time specified by the jmGeneralJobPersistence object, rather than the (shorter) jmGeneralAttributePersistence object.

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 attribute of the first document in the job or the fileName attribute 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.

jobServiceTypes(24), JmJobServiceTypesTC
INTEGER: Specifies the type(s) of service to which the job has been submitted (print, fax, scan, etc.). The service type is bit encoded with each job service type so that more general and arbitrary services can be created, such as services with more than one destination type, or ones with only a source or only a destination. For example, a job service might scan, faxOut, and print a single job. In this case, three bits would be set in the jobServiceTypes attribute, corresponding to the hexadecimal values: 0x8 + 0x20 + 0x4, respectively, yielding: 0x2C.

Whether this attribute is set from a job attribute supplied by the job submission client or is set by the recipient job submission server or device depends on the job submission protocol. This attribute SHALL be implemented if the server or device has other types in addition to or instead of printing.

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

jobSourceChannelIndex(25), Integer32 (0..2147483647)
INTEGER: The index of the row in the associated Printer MIB[print-mib] of the channel which is the source of the print job.

jobSourcePlatformType(26), JmJobSourcePlatformTypeTC
INTEGER: The source platform type of the immediate upstream submitter that submitted the job to the server (configuration 2) or device (configuration 1 and 3) to which the agent is providing access. For configuration 1, this is the type of the client that submitted the job to the device; for configuration 2, this is the type of the client that submitted the job to the server; and for configuration 3, this is the type of the server that submitted the job to the device.

submittingServerName(27), JmJobStringTC (SIZE(0..63))
OCTETS: For configuration 3 only: The administrative name of the server that submitted the job to the device.

submittingApplicationName(28), JmJobStringTC (SIZE(0..63))
OCTETS: The name of the client application (not the server in configuration 3) that submitted the job to the server or device.

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

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

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

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

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

AND/OR

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

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

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

fileName(34), JmJobStringTC (SIZE(0..63))
OCTETS: MULTI-ROW: The coded character set file name or URI[URI-spec] of the document.

There is no restriction on the same file name occurring in multiple rows.

documentName(35), JmJobStringTC (SIZE(0..63))
OCTETS: MULTI-ROW: The coded character set name of the document.

There is no restriction on the same document name occurring in multiple rows.

jobComment(36), JmJobStringTC (SIZE(0..63))
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.

The jobComment attribute is not intended to be a name; see the jobName attribute.

documentFormatIndex(37), Integer32 (0..2147483647)
INTEGER: MULTI-ROW: The index in the prtInterpreterTable in the Printer MIB[print-mib] of the page description language (PDL) or control language interpreter that this job requires/uses. A document or a job MAY use more than one PDL or control language.

NOTE - As with all intensive attributes where multiple rows are allowed, there SHALL be only one distinct row for each distinct interpreter; 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. Such an agent SHALL use the documentFormat attribute instead.

```
documentFormat(38),                PrtInterpreterLangFamilyTC
                                   AND/OR
                                   OCTET STRING(SIZE(0..63))
    INTEGER:  MULTI-ROW:  The interpreter language family
    corresponding to the Printer MIB[print-mib]
    prtInterpreterLangFamily object, that this job
    requires/uses.  A document or a job MAY use more than one
    PDL or control language.

    AND/OR

    OCTETS:  MULTI-ROW:  The document format registered as a
    media type[iana-media-types], i.e., the name of the MIME
    content-type/subtype.  Examples: 'application/postscript',
    'application/vnd.hp-PCL', 'application/pdf', 'text/plain'
    (US-ASCII SHALL be assumed), 'text/plain; charset=iso-8859-
    1', and 'application/octet-stream'.  The IPP 'document-
    format' job attribute uses these same values with the same
    semantics.  See the IPP [ipp-model] 'mimeMediaType'
    attribute syntax and the document-format attribute for
    further examples and explanation.


++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ Job Parameter attributes
+
+ The following attributes represent input parameters
+ supplied by the submitting client in the job submission
+ protocol.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

jobPriority(50),                   Integer32 (-2..100)
    INTEGER:  The priority for 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.  -2 indicates
    unknown.
```

jobProcessAfterDateAndTime(51), DateAndTime (SNMPv2-TC)
OCTETS: 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 jmJobState object to pendingHeld and add the jobProcessAfterSpecified bit value to the job's jmJobStateReasons1 object. When the specified date and time arrives, the server SHALL remove the jobProcessAfterSpecified bit value from the job's jmJobStateReasons1 object and, if no other reasons remain, SHALL change the job's jmJobState object to pending.

jobHold(52), JmBooleanTC
INTEGER: If the value is 'true(4)', a client has explicitly specified that the job is to be held until explicitly released. Until the job is explicitly released by a client, the job SHALL be in the pendingHeld state with the jobHoldSpecified value in the jmJobStateReasons1 attribute.

jobHoldUntil(53), JmJobStringTC (SIZE(0..63))
OCTETS: The named time period during which the job SHALL become a candidate for processing, such as 'evening', 'night', 'weekend', 'second-shift', 'third-shift', etc., as defined by the system administrator. See IPP [ipp-model] for the standard keyword values. Until that time period arrives, the job SHALL be in the pendingHeld state with the jobHoldUntilSpecified value in the jmJobStateReasons1 object. The value 'no-hold' SHALL indicate explicitly that no time period has been specified; the absence of this attribute SHALL indicate implicitly that no time period has been specified.

outputBin(54), Integer32 (0..2147483647)
AND/OR
JmJobStringTC (SIZE(0..63))
INTEGER: MULTI-ROW: The output subunit index in the Printer MIB[print-mib]

AND/OR

OCTETS: MULTI-ROW: the name or number (represented as ASCII digits) of the output bin to which all or part of the job is placed in.

```
2431          sides(55),                          Integer32 (-2..2)
2432              INTEGER:  MULTI-ROW:  The number of sides, '1' or '2', that
2433              any document in this job requires/used.
2434
2435          finishing(56),                      JmFinishingTC
2436              INTEGER:  MULTI-ROW:  Type of finishing that any document
2437              in this job requires/used.
2438
2439
2440          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2441          + Image Quality attributes (requested and consumed)
2442          +
2443          + For devices that can vary the image quality.
2444          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2445
2446          printQualityRequested(70),          JmPrintQualityTC
2447              INTEGER:  MULTI-ROW:  The print quality selection requested
2448              for a document in the job for printers that allow quality
2449              differentiation.
2450
2451          printQualityUsed(71),               JmPrintQualityTC
2452              INTEGER:  MULTI-ROW:  The print quality selection actually
2453              used by a document in the job for printers that allow
2454              quality differentiation.
2455
2456          printerResolutionRequested(72),     JmPrinterResolutionTC
2457              OCTETS:  MULTI-ROW:  The printer resolution requested for a
2458              document in the job for printers that support resolution
2459              selection.
2460
2461          printerResolutionUsed(73),          JmPrinterResolutionTC
2462              OCTETS:  MULTI-ROW:  The printer resolution actually used
2463              by a document in the job for printers that support
2464              resolution selection.
2465
2466          tonerEcomonyRequested(74),          JmTonerEconomyTC
2467              INTEGER:  MULTI-ROW:  The toner economy selection requested
2468              for documents in the job for printers that allow toner
2469              economy differentiation.
2470
2471          tonerEcomonyUsed(75),               JmTonerEconomyTC
2472              INTEGER:  MULTI-ROW:  The toner economy selection actually
2473              used by documents in the job for printers that allow toner
2474              economy differentiation.
2475
2476          tonerDensityRequested(76),          Integer32 (-2..100)
2477              INTEGER:  MULTI-ROW:  The toner density requested for a
2478              document in this job for devices that can vary toner
2479              density levels.  Level 1 is the lowest density and level
2480              100 is the highest density level.  Devices with a smaller
2481              range, SHALL map the 1-100 range evenly onto the
2482              implemented range.
```

```
2483
2484          tonerDensityUsed(77),              Integer32 (-2..100)
2485              INTEGER:  MULTI-ROW:  The toner density used by documents
2486              in this job for devices that can vary toner density levels.
2487              Level 1 is the lowest density and level 100 is the highest
2488              density level.  Devices with a smaller range, SHALL map the
2489              1-100 range evenly onto the implemented range.
2490
2491
2492          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2493          + Job Progress attributes (requested and consumed)
2494          +
2495          + Pairs of these attributes can be used by monitoring
2496          + applications to show an indication of relative progress
2497          + to users.  See section 3.4, entitled
2498          + 'Monitoring Job Progress'.
2499          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2500
2501          jobCopiesRequested(90),            Integer32 (-2..2147483647)
2502              INTEGER:  The number of copies of the entire job that are
2503              to be produced.
2504
2505          jobCopiesCompleted(91),            Integer32 (-2..2147483647)
2506              INTEGER:  The number of copies of the entire job that have
2507              been completed so far.
2508
2509          documentCopiesRequested(92),       Integer32 (-2..2147483647)
2510              INTEGER:  The total count of the number of document copies
2511              requested for the job as a whole.  If there are documents
2512              A, B, and C, and document B is specified to produce 4
2513              copies, the number of document copies requested is 6 for
2514              the job.
2515
2516              This attribute SHALL be used only when a job has multiple
2517              documents.  The jobCopiesRequested attribute SHALL be used
2518              when the job has only one document.
2519
2520          documentCopiesCompleted(93),       Integer32 (-2..2147483647)
2521              INTEGER:  The total count of the number of document copies
2522              completed so far for the job as a whole.  If there are
2523              documents A, B, and C, and document B is specified to
2524              produce 4 copies, the number of document copies starts a 0
2525              and runs up to 6 for the job as the job processes.
2526
2527              This attribute SHALL be used only when a job has multiple
2528              documents.  The jobCopiesCompleted attribute SHALL be used
2529              when the job has only one document.
2530
```

```
2531          jobKOctetsTransferred(94),        Integer32 (-2..2147483647)
2532              INTEGER:  The number of K (1024) octets transferred to the
2533              server or device to which the agent is providing access.
2534              This count is independent of the number of copies of the
2535              job or documents that will be produced, but it is only a
2536              measure of the number of bytes transferred to the server or
2537              device.
2538
2539              The agent SHALL round the actual number of octets
2540              transferred up to the next higher K.  Thus 0 octets SHALL
2541              be represented as '0', 1-1024 octets SHALL BE represented
2542              as '1', 1025-2048 SHALL be '2', etc.  When the job
2543              completes, the values of the jmJobKOctetsPerCopyRequested
2544              object and the jobKOctetsTransferred attribute SHALL be
2545              equal.
2546
2547              NOTE - The jobKOctetsTransferred can be used with the
2548              jmJobKOctetsPerCopyRequested object in order to produce a
2549              relative indication of the progress of the job for agents
2550              that do not implement the jmJobKOctetsProcessed object.
2551
2552          sheetCompletedCopyNumber(95),     Integer32 (-2..2147483647)
2553              INTEGER:  The number of the copy being stacked for the
2554              current document.  This number starts at 0, is set to 1
2555              when the first sheet of the first copy for each document is
2556              being stacked and is equal to n where n is the nth sheet
2557              stacked in the current document copy.  See section 3.4 ,
2558              entitled 'Monitoring Job Progress'.
2559
2560          sheetCompletedDocumentNumber(96), Integer32 (-2..2147483647)
2561              INTEGER:  The ordinal number of the document in the job
2562              that is currently being stacked.  This number starts at 0,
2563              increments to 1 when the first sheet of the first document
2564              in the job is being stacked, and is equal to n where n is
2565              the nth document in the job, starting with 1.
2566
2567              Implementations that only support one document jobs SHOULD
2568              NOT implement this attribute.
2569
2570          jobCollationType(97),             JmJobCollationTypeTC
2571              INTEGER:  The type of job collation. See also Section 3.4,
2572              entitled 'Monitoring Job Progress'.
2573
```

```
2574
2575          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2576          + Impression attributes
2577          +
2578          + See the definition of the terms 'impression', 'sheet',
2579          + and 'page' in Section 2.
2580          +
2581          + See also jmJobImpressionsPerCopyRequested and
2582          + jmJobImpressionsCompleted objects in the jmJobTable.
2583          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2584
2585          impressionsSpooled(110),          Integer32 (-2..2147483647)
2586              INTEGER:  The number of impressions spooled to the server
2587              or device for the job so far.
2588
2589          impressionsSentToDevice(111),     Integer32 (-2..2147483647)
2590              INTEGER:  The number of impressions sent to the device for
2591              the job so far.
2592
2593          impressionsInterpreted(112),      Integer32 (-2..2147483647)
2594              INTEGER:  The number of impressions interpreted for the job
2595              so far.
2596
2597          impressionsCompletedCurrentCopy(113),
2598                                            Integer32 (-2..2147483647)
2599              INTEGER:  The number of impressions completed by the device
2600              for the current copy of the current document so far.  For
2601              printing, the impressions completed includes interpreting,
2602              marking, and stacking the output.  For other types of job
2603              services, the number of impressions completed includes the
2604              number of impressions processed.
2605
2606              This value SHALL be reset to 0 for each document in the job
2607              and for each document copy.
2608
2609          fullColorImpressionsCompleted(114), Integer32 (-2..2147483647)
2610              INTEGER:  The number of full color impressions completed by
2611              the device for this job so far.  For printing, the
2612              impressions completed includes interpreting, marking, and
2613              stacking the output.  For other types of job services, the
2614              number of impressions completed includes the number of
2615              impressions processed. Full color impressions are typically
2616              defined as those requiring 3 or more colorants, but this
2617              MAY vary by implementation.  In any case, the value of this
2618              attribute counts by 1 for each side that has full color,
2619              not by the number of colors per side (and the other
2620              impression counters are incremented, except
2621              highlightColorImpressionsCompleted(115)).
2622
```

```
2623          highlightColorImpressionsCompleted(115),
2624                                          Integer32 (-2..2147483647)
2625              INTEGER:  The number of highlight color impressions
2626              completed by the device for this job so far.  For printing,
2627              the impressions completed includes interpreting, marking,
2628              and stacking the output.  For other types of job services,
2629              the number of impressions completed includes the number of
2630              impressions processed.  Highlight color impressions are
2631              typically defined as those requiring black plus one other
2632              colorant, but this MAY vary by implementation.  In any
2633              case, the value of this attribute counts by 1 for each side
2634              that has highlight color (and the other impression counters
2635              are incremented, except
2636              fullColorImpressionsCompleted(114)).
2637
2638
2639          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2640          + Page attributes
2641          +
2642          + See the definition of 'impression', 'sheet', and 'page'
2643          + in Section 2.
2644          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2645
2646          pagesRequested(130),                Integer32 (-2..2147483647)
2647              INTEGER:  The number of logical pages requested by the job
2648              to be processed.
2649
2650          pagesCompleted(131),                Integer32 (-2..2147483647)
2651              INTEGER:  The number of logical pages completed for this
2652              job so far.
2653
2654              For implementations where multiple copies are produced by
2655              the interpreter with only a single pass over the data, the
2656              final value SHALL be equal to the value of the
2657              pagesRequested object.  For implementations where multiple
2658              copies are produced by the interpreter by processing the
2659              data for each copy, the final value SHALL be a multiple of
2660              the value of the pagesRequested object.
2661
2662              NOTE - See the impressionsCompletedCurrentCopy and
2663              pagesCompletedCurrentCopy attributes for attributes that
2664              are reset on each document copy.
2665
2666              NOTE - The pagesCompleted object can be used with the
2667              pagesRequested object to provide an indication of the
2668              relative progress of the job, provided that the
2669              multiplicative factor is taken into account for some
2670              implementations of multiple copies.
2671
```

```
2672          pagesCompletedCurrentCopy(132),   Integer32 (-2..2147483647)
2673              INTEGER:  The number of logical pages completed for the
2674              current copy of the document so far.  This value SHALL be
2675              reset to 0 for each document in the job and for each
2676              document copy.
2677
2678
2679          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2680          + Sheet attributes
2681          +
2682          + See the definition of 'impression', 'sheet', and 'page'
2683          + in Section 2.
2684          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2685
2686          sheetsRequested(150),             Integer32 (-2..2147483647)
2687              INTEGER:  The total number of medium sheets requested to be
2688              produced for this job.
2689
2690              Unlike the jmJobKOctetsPerCopyRequested and
2691              jmJobImpressionsPerCopyRequested attributes, the
2692              sheetsRequested(150) attribute SHALL include the
2693              multiplicative factor contributed by the number of copies
2694              and so is the total number of sheets to be produced by the
2695              job, as opposed to the size of the document(s) submitted.
2696
2697          sheetsCompleted(151),             Integer32 (-2..2147483647)
2698              INTEGER:  The total number of medium sheets that have
2699              completed marking and stacking for the entire job so far
2700              whether those sheets have been processed on one side or on
2701              both.
2702
2703          sheetsCompletedCurrentCopy(152),  Integer32 (-2..2147483647)
2704              INTEGER:  The number of medium sheets that have completed
2705              marking and stacking for the current copy of a document in
2706              the job so far whether those sheets have been processed on
2707              one side or on both.
2708
2709              The value of this attribute SHALL be 0 before the job
2710              starts processing and SHALL be reset to 1 after the first
2711              sheet of each document and document copy in the job is
2712              processed and stacked.
2713
2714
```

```
2715          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2716          + Resources attributes (requested and consumed)
2717          +
2718          + Pairs of these attributes can be used by monitoring
2719          + applications to show an indication of relative usage to
2720          + users.
2721          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2722
2723          mediumRequested(170),              JmMediumTypeTC
2724                                             AND/OR
2725                                             JmJobStringTC (SIZE(0..63))
2726              INTEGER:  MULTI-ROW:  The type
2727              AND/OR
2728              OCTETS:  MULTI-ROW:  the name of the medium that is
2729              required by the job.
2730
2731              NOTE - The name (JmJobStringTC) values correspond to the
2732              prtInputMediaName object in the Printer MIB [print-mib] and
2733              the values of the IPP 'media' attribute.
2734
2735          mediumConsumed(171),               Integer32 (-2..2147483647)
2736                                             AND
2737                                             JmJobStringTC (SIZE(0..63))
2738              INTEGER:  MULTI-ROW:  The number of sheets
2739              AND
2740              OCTETS:  MULTI-ROW:  the name of the medium that has been
2741              consumed so far whether those sheets have been processed on
2742              one side or on both.
2743
2744              This attribute SHALL have both Integer32 and OCTET STRING
2745              (represented as  JmJobStringTC) values.
2746
2747              NOTE - The name (JmJobStringTC) values correspond to the
2748              name values of the prtInputMediaName object in the Printer
2749              MIB [print-mib].
2750
2751          colorantRequested(172),            Integer32 (-2..2147483647)
2752                                             AND/OR
2753                                             JmJobStringTC (SIZE(0..63))
2754              INTEGER:  MULTI-ROW:  The index (prtMarkerColorantIndex) in
2755              the Printer MIB[print-mib]
2756              AND/OR
2757              OCTETS:  MULTI-ROW:  the name of the colorant requested.
2758
2759              NOTE - The name (JmJobStringTC) values correspond to the
2760              name values of the prtMarkerColorantValue object in the
2761              Printer MIB.  Examples are: red, blue.
```

```
2762          colorantConsumed(173),              Integer32 (-2..2147483647)
2763                                              AND/OR
2764                                              JmJobStringTC (SIZE(0..63))
2765              INTEGER:  MULTI-ROW:  The index (prtMarkerColorantIndex) in
2766              the Printer MIB[print-mib]
2767              AND/OR
2768              OCTETS:  MULTI-ROW:  the name of the colorant consumed.
2769
2770              NOTE - The name (JmJobStringTC) values correspond to the
2771              name values of the prtMarkerColorantValue object in the
2772              Printer MIB.  Examples are: red, blue
2773
2774
2775          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2776          + Time attributes (set by server or device)
2777          +
2778          + This section of attributes are ones that are set by the
2779          + server or device that accepts jobs.  Two forms of time are
2780          + provided.  Each form is represented in a separate attribute.
2781          + See section 3.1.2 and section 3.1.3 for the
2782          + conformance requirements for time attribute for agents and
2783          + monitoring applications, respectively.  The two forms are:
2784          +
2785          + 'DateAndTime' is an 8 or 11 octet binary encoded year,
2786          + month, day, hour, minute, second, deci-second with
2787          + optional offset from UTC.  See SNMPv2-TC [SMIv2-TC].
2788          +
2789          + NOTE: 'DateAndTime' is not printable characters; it is
2790          + binary.
2791          +
2792          + 'JmTimeStampTC' is the time of day measured in the number of
2793          + seconds since the system was booted.
2794          ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
2795
2796          jobSubmissionToServerTime(190),   JmTimeStampTC
2797                                            AND/OR
2798                                            DateAndTime
2799              INTEGER:  Configuration 3 only:  The time
2800              AND/OR
2801              OCTETS:  the date and time that the job was submitted to
2802              the server (as distinguished from the device which uses
2803              jobSubmissionTime).
2804
2805          jobSubmissionTime(191),           JmTimeStampTC
2806                                            AND/OR
2807                                            DateAndTime
2808              INTEGER:  Configurations 1, 2, and 3:  The time
2809              AND/OR
2810              OCTETS:  the date and time that the job was submitted to
2811              the server or device to which the agent is providing
2812              access.
2813
```

```
2814          jobStartedBeingHeldTime(192),     JmTimeStampTC
2815                                            AND/OR
2816                                            DateAndTime
2817              INTEGER:  The time
2818              AND/OR
2819              OCTETS:  the date and time that the job last entered the
2820              pendingHeld state.  If the job has never entered the
2821              pendingHeld state, then the value SHALL be '0' or the
2822              attribute SHALL not be present in the table.
2823
2824          jobStartedProcessingTime(193),    JmTimeStampTC
2825                                            AND/OR
2826                                            DateAndTime
2827              INTEGER:  The time
2828              AND/OR
2829              OCTETS:  the date and time that the job started processing.
2830
2831          jobCompletionTime(194),           JmTimeStampTC
2832                                            AND/OR
2833                                            DateAndTime
2834              INTEGER:  The time
2835              AND/OR
2836              OCTETS:  the date and time that the job entered the
2837              completed, canceled, or aborted state.
2838
2839          jobProcessingCPUTime(195)         Integer32 (-2..2147483647)
2840              UNITS     'seconds'
2841              INTEGER:  The amount of CPU time in seconds that the job
2842              has been in the processing state.  If the job enters the
2843              processingStopped state, that elapsed time SHALL not be
2844              included.  In other words, the jobProcessingCPUTime value
2845              SHOULD be relatively repeatable when the same job is
2846              processed again on the same device."
2847
2848      REFERENCE
2849          "See Section 3.2 entitled 'The Attribute Mechanism' for a
2850          description of this textual-convention and its use in the
2851          jmAttributeTable.
2852
2853          This is a type 2 enumeration.  See Section 3.7.1.2."
2854      SYNTAX      INTEGER {
2855          other(1),
2856
2857          -- Job State attributes:
2858          jobStateReasons2(3),
2859          jobStateReasons3(4),
2860          jobStateReasons4(5),
2861          processingMessage(6),
2862          processingMessageNaturalLangTag(7),
2863          jobCodedCharSet(8),
2864          jobNaturalLanguageTag(9),
2865
```

```
2866            -- Job Identification attributes:
2867            jobURI(20),
2868            jobAccountName(21),
2869            serverAssignedJobName(22),
2870            jobName(23),
2871            jobServiceTypes(24),
2872            jobSourceChannelIndex(25),
2873            jobSourcePlatformType(26),
2874            submittingServerName(27),
2875            submittingApplicationName(28),
2876            jobOriginatingHost(29),
2877            deviceNameRequested(30),
2878            queueNameRequested(31),
2879            physicalDevice(32),
2880            numberOfDocuments(33),
2881            fileName(34),
2882            documentName(35),
2883            jobComment(36),
2884            documentFormatIndex(37),
2885            documentFormat(38),
2886
2887            -- Job Parameter attributes:
2888            jobPriority(50),
2889            jobProcessAfterDateAndTime(51),
2890            jobHold(52),
2891            jobHoldUntil(53),
2892            outputBin(54),
2893            sides(55),
2894            finishing(56),
2895
2896            -- Image Quality attributes:
2897            printQualityRequested(70),
2898            printQualityUsed(71),
2899            printerResolutionRequested(72),
2900            printerResolutionUsed(73),
2901            tonerEcomonyRequested(74),
2902            tonerEcomonyUsed(75),
2903            tonerDensityRequested(76),
2904            tonerDensityUsed(77),
2905
2906            -- Job Progress attributes:
2907            jobCopiesRequested(90),
2908            jobCopiesCompleted(91),
2909            documentCopiesRequested(92),
2910            documentCopiesCompleted(93),
2911            jobKOctetsTransferred(94),
2912            sheetCompletedCopyNumber(95),
2913            sheetCompletedDocumentNumber(96),
2914            jobCollationType(97),
2915
```

```
2916          -- Impression attributes:
2917          impressionsSpooled(110),
2918          impressionsSentToDevice(111),
2919          impressionsInterpreted(112),
2920          impressionsCompletedCurrentCopy(113),
2921          fullColorImpressionsCompleted(114),
2922          highlightColorImpressionsCompleted(115),
2923
2924          -- Page attributes:
2925          pagesRequested(130),
2926          pagesCompleted(131),
2927          pagesCompletedCurrentCopy(132),
2928
2929          -- Sheet attributes:
2930          sheetsRequested(150),
2931          sheetsCompleted(151),
2932          sheetsCompletedCurrentCopy(152),
2933
2934          -- Resource attributes:
2935          mediumRequested(170),
2936          mediumConsumed(171),
2937          colorantRequested(172),
2938          colorantConsumed(173),
2939
2940          -- Time attributes:
2941          jobSubmissionToServerTime(190),
2942          jobSubmissionTime(191),
2943          jobStartedBeingHeldTime(192),
2944          jobStartedProcessingTime(193),
2945          jobCompletionTime(194),
2946          jobProcessingCPUTime(195)
2947      }
2948
2949
2950
2951
```

```
JmJobServiceTypesTC ::= TEXTUAL-CONVENTION
    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, faxOut, and print a single job.  In
        this case, three bits would be set in the jobServiceTypes
        attribute, corresponding to the hexadecimal values: 0x8 + 0x20
        + 0x4, respectively, yielding: 0x2C.

        Whether this attribute is set from a job attribute supplied by
        the job submission client or is set by the recipient job
        submission server or device depends on the job submission
        protocol.  With either implementation, the agent SHALL return a
        non-zero value for this attribute indicating the type of the
        job.

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

        The following service component types are defined (in
        hexadecimal) and are assigned a separate bit value for use with
        the jobServiceTypes attribute:

        other                                0x1
            The job contains some instructions that are not one of the
            identified types.

        unknown                              0x2
            The job contains some instructions whose type is unknown to
            the agent.

        print                                0x4
            The job contains some instructions that specify printing

        scan                                 0x8
            The job contains some instructions that specify scanning

        faxIn                                0x10
            The job contains some instructions that specify receive fax

        faxOut                               0x20
            The job contains some instructions that specify sending fax
```

```
3003            getFile                             0x40
3004                The job contains some instructions that specify accessing
3005                files or documents
3006
3007            putFile                             0x80
3008                The job contains some instructions that specify storing
3009                files or documents
3010
3011            mailList                            0x100
3012                The job contains some instructions that specify
3013                distribution of documents using an electronic mail system."
3014        REFERENCE
3015            "These bit definitions are the equivalent of a type 2 enum
3016            except that combinations of them MAY be used together.  See
3017            section 3.7.1.2."
3018        SYNTAX      INTEGER (0..2147483647)   -- 31 bits, all but sign bit
3019
3020
3021
3022    JmJobStateReasons1TC ::= TEXTUAL-CONVENTION
3023        STATUS      current
3024        DESCRIPTION
3025            "The JmJobStateReasonsNTC (N=1..4) textual-conventions are used
3026            with the jmJobStateReasons1 object and jobStateReasonsN
3027            (N=2..4), respectively, to provide additional information
3028            regarding the current jmJobState object value.  These values
3029            MAY be used with any job state or states for which the reason
3030            makes sense.
3031
3032            NOTE - While values cannot be added to the jmJobState object
3033            without impacting deployed clients that take actions upon
3034            receiving jmJobState values, it is the intent that additional
3035            JmJobStateReasonsNTC enums can be defined and registered
3036            without impacting such deployed clients.  In other words, the
3037            jmJobStateReasons1 object and jobStateReasonsN attributes are
3038            intended to be extensible.
3039
3040            NOTE - The Job Monitoring MIB contains a superset of the IPP
3041            values[ipp-model] for the IPP 'job-state-reasons' attribute,
3042            since the Job Monitoring MIB is intended to cover other job
3043            submission protocols as well.  Also some of the names of the
3044            reasons have been changed from 'printer' to 'device', since the
3045            Job Monitoring MIB is intended to cover additional types of
3046            devices, including input devices, such as scanners.
3047
3048            The following standard values are defined (in hexadecimal) as
3049            powers of two, since multiple values MAY be used at the same
3050            time.  For ease of understanding, the JmJobStateReasons1TC
3051            reasons are presented in the order in which the reasons are
3052            likely to occur (if implemented), starting with the
3053            'jobIncoming' value and ending with the
3054            'jobCompletedWithErrors' value.
```

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

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

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

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

jobInterpreting 0x800
The device to which the job is assigned is interpreting the document data.

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

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

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

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

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

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

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

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

jobCompletedSuccessfully 0x80000
The job completed successfully.

jobCompletedWithWarnings 0x100000
The job completed with warnings.

jobCompletedWithErrors 0x200000
The job completed with errors (and possibly warnings too).

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

jobPaused 0x400000
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 agent SHALL remove the jobPaused values from the job's jmJobStateReasons1 object and the job is eventually resumed at or near the point where the job was paused.

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

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

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

connectingToDeviceTimeOut 0x1000
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.

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

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

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

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

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

validating 0x40000
The server/device is validating the job *after* accepting the job.

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

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

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

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

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

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

```
jmGeneralJobSetIndex OBJECT-TYPE
    SYNTAX      Integer32 (1..32767)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A unique value for each job set in this MIB.  The jmJobTable
        and jmAttributeTable tables have this same index as their
        primary index.

        The value(s) of the jmGeneralJobSetIndex SHALL be persistent
        across power cycles, so that clients that have retained
        jmGeneralJobSetIndex values will access the same job sets upon
        subsequent power-up.

        An implementation that has only one job set, such as a printer
        with a single queue, SHALL hard code this object with the value
        1."
    REFERENCE
        "See Section 2 entitled 'Terminology and Job Model' for the
        definition of a job set.
        Corresponds to the first index in jmJobTable and
        jmAttributeTable."
    ::= { jmGeneralEntry 1 }


jmGeneralNumberOfActiveJobs OBJECT-TYPE
    SYNTAX      Integer32 (0..2147483647)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The current number of 'active' jobs in the jmJobIDTable,
        jmJobTable, and jmAttributeTable, i.e., the total number of
        jobs that are in the pending, processing, or processingStopped
        states.  See the JmJobStateTC textual-convention for the exact
        specification of the semantics of the job states."
    DEFVAL      { 0 }      -- no jobs
    ::= { jmGeneralEntry 2 }
```

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

```
jmGeneralJobPersistence OBJECT-TYPE
    SYNTAX      Integer32 (15..2147483647)
    UNITS       "seconds"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The minimum time in seconds for this instance of the Job Set
        that an entry SHALL remain in the jmJobIDTable and jmJobTable
        after processing has completed, i.e., the minimum time in
        seconds starting when the job enters the completed, canceled,
        or aborted state.

        Configuring this object is implementation-dependent.

        This value SHALL be equal to or greater than the value of
        jmGeneralAttributePersistence.  This value SHOULD be at least
        60 which gives a monitoring application one minute in which to
        poll for job data."
    DEFVAL      { 60 }          -- one minute
    ::= { jmGeneralEntry 5 }


jmGeneralAttributePersistence OBJECT-TYPE
    SYNTAX      Integer32 (15..2147483647)
    UNITS       "seconds"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The minimum time in seconds for this instance of the Job Set
        that an entry SHALL remain in the jmAttributeTable after
        processing has completed , i.e., the time in seconds starting
        when the job enters the completed, canceled, or aborted state.

        Configuring this object is implementation-dependent.

        This value SHOULD be at least 60 which gives a monitoring
        application one minute in which to poll for job data."
    DEFVAL      { 60 }          -- one minute
    ::= { jmGeneralEntry 6 }
```

```
jmGeneralJobSetName OBJECT-TYPE
    SYNTAX      JmUTF8StringTC (SIZE(0..63))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The human readable name of this job set assigned by the system
        administrator (by means outside of this MIB).  Typically, this
        name SHOULD be the name of the job queue.  If a server or
        device has only a single job set, this object can be the
        administratively assigned name of the server or device itself.
        This name does not need to be unique, though each job set in a
        single Job Monitoring MIB SHOULD have distinct names.

        NOTE - If the job set corresponds to a single printer and the
        Printer MIB is implemented, this value SHOULD be the same as
        the prtGeneralPrinterName object in the draft Printer MIB
        [print-mib-draft].  If the job set corresponds to an IPP
        Printer, this value SHOULD be the same as the IPP 'printer-
        name' Printer attribute.

        NOTE - The purpose of this object is to help the user of the
        job monitoring application distinguish between several job sets
        in implementations that support more than one job set."
    REFERENCE
        "See the OBJECT compliance macro for the minimum maximum length
        required for conformance."
    DEFVAL      { ''H }      -- empty string
    ::= { jmGeneralEntry 7 }
```

```
3658  -- The Job ID Group (MANDATORY)
3659
3660  -- The jmJobIDGroup consists entirely of the jmJobIDTable.
3661
3662  jmJobID  OBJECT IDENTIFIER ::= { jobmonMIBObjects 2 }
3663
3664  jmJobIDTable  OBJECT-TYPE
3665      SYNTAX      SEQUENCE OF JmJobIDEntry
3666      MAX-ACCESS  not-accessible
3667      STATUS      current
3668      DESCRIPTION
3669          "The jmJobIDTable provides a correspondence map (1) between the
3670          job submission ID that a client uses to refer to a job and (2)
3671          the jmGeneralJobSetIndex and jmJobIndex that the Job Monitoring
3672          MIB agent assigned to the job and that are used to access the
3673          job in all of the other tables in the MIB.  If a monitoring
3674          application already knows the jmGeneralJobSetIndex and the
3675          jmJobIndex of the job it is querying, that application NEED NOT
3676          use the jmJobIDTable."
3677      REFERENCE
3678          "The MANDATORY-GROUP macro specifies that this group is
3679          MANDATORY."
3680      ::= { jmJobID 1 }
3681
3682
3683
3684  jmJobIDEntry  OBJECT-TYPE
3685      SYNTAX      JmJobIDEntry
3686      MAX-ACCESS  not-accessible
3687      STATUS      current
3688      DESCRIPTION
3689          "The map from (1) the jmJobSubmissionID to (2) the
3690          jmGeneralJobSetIndex and jmJobIndex.
3691
3692          An entry SHALL exist in this table for each job currently known
3693          to the agent for all job sets and job states.  There MAY be
3694          more than one jmJobIDEntry that maps to a single job.  This
3695          many to one mapping can occur when more than one network entity
3696          along the job submission path supplies a job submission ID.
3697          See Section 3.5.  However, each job SHALL appear once and in
3698          one and only one job set."
3699      INDEX  { jmJobSubmissionID }
3700      ::= { jmJobIDTable 1 }
3701
3702  JmJobIDEntry ::= SEQUENCE {
3703      jmJobSubmissionID                   OCTET STRING(SIZE(48)),
3704      jmJobIDJobSetIndex                  Integer32 (0..32767),
3705      jmJobIDJobIndex                     Integer32 (0..2147483647)
3706  }
3707
```

```
jmJobSubmissionID OBJECT-TYPE
    SYNTAX      OCTET STRING(SIZE(48))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A quasi-unique 48-octet fixed-length string ID which
        identifies the job within a particular client-server
        environment.  There are multiple formats for the
        jmJobSubmissionID.  Each format SHALL be uniquely identified.
        See the JmJobSubmissionIDTypeTC textual convention.  Each
        format SHALL be registered using the procedures of a type 2
        enum.  See section 3.7.3 entitled: 'PWG Registration of Job
        Submission Id Formats'.

        If the requester (client or server) does not supply a job
        submission ID in the job submission protocol, then the
        recipient (server or device) SHALL assign a job submission ID
        using any of the standard formats that have been reserved for
        agents and adding the final 8 octets to distinguish the ID from
        others submitted from the same requester.

        The monitoring application, whether in the client or running
        separately, MAY use the job submission ID to help identify
        which jmJobIndex was assigned by the agent, i.e., in which row
        the job information is in the other tables.

        NOTE - fixed-length is used so that a management application
        can use a shortened GetNext varbind (in SNMPv1 and SNMPv2) in
        order to get the next submission ID, disregarding the remainder
        of the ID in order to access jobs independent of the trailing
        identifier part, e.g., to get all jobs submitted by a
        particular jmJobOwner or submitted from a particular MAC
        address."
    REFERENCE
        "See the JmJobSubmissionIDTypeTC textual convention.
        See APPENDIX B - Support of Job Submission Protocols."
    ::= { jmJobIDEntry 1 }
```

```
jmJobIDJobSetIndex OBJECT-TYPE
    SYNTAX      Integer32 (0..32767)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object contains the value of the jmGeneralJobSetIndex for
        the job with the jmJobSubmissionID value, i.e., the job set
        index of the job set in which the job was placed when that
        server or device accepted the job.  This 16-bit value in
        combination with the jmJobIDJobIndex value permits the
        management application to access the other tables to obtain the
        job-specific objects for this job."
    REFERENCE
        "See jmGeneralJobSetIndex in the jmGeneralTable."
    DEFVAL      { 0 }      -- 0 indicates no job set index
    ::= { jmJobIDEntry 2 }


jmJobIDJobIndex OBJECT-TYPE
    SYNTAX      Integer32 (0..2147483647)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object contains the value of the jmJobIndex for the job
        with the jmJobSubmissionID value, i.e., the job index for the
        job when the server or device accepted the job.  This value, in
        combination with the jmJobIDJobSetIndex value, permits the
        management application to access the other tables to obtain the
        job-specific objects for this job."
    REFERENCE
        "See jmJobIndex in the jmJobTable."
    DEFVAL      { 0 }      -- 0 indicates no jmJobIndex value.
    ::= { jmJobIDEntry 3 }
```

```
3784  -- The Job Group (MANDATORY)
3785
3786  -- The jmJobGroup consists entirely of the jmJobTable.
3787
3788  jmJob  OBJECT IDENTIFIER ::= { jobmonMIBObjects 3 }
3789
3790  jmJobTable  OBJECT-TYPE
3791      SYNTAX      SEQUENCE OF JmJobEntry
3792      MAX-ACCESS  not-accessible
3793      STATUS      current
3794      DESCRIPTION
3795          "The jmJobTable consists of basic job state and status
3796          information for each job in a job set that (1) monitoring
3797          applications need to be able to access in a single SNMP Get
3798          operation, (2) that have a single value per job, and (3) that
3799          SHALL always be implemented."
3800      REFERENCE
3801          "The MANDATORY-GROUP macro specifies that this group is
3802          MANDATORY."
3803      ::= { jmJob 1 }
3804
3805
3806
3807  jmJobEntry  OBJECT-TYPE
3808      SYNTAX      JmJobEntry
3809      MAX-ACCESS  not-accessible
3810      STATUS      current
3811      DESCRIPTION
3812          "Basic per-job state and status information.
3813
3814          An entry SHALL exist in this table for each job, no matter what
3815          the state of the job is.  Each job SHALL appear in one and only
3816          one job set."
3817      REFERENCE
3818          "See Section 3.2 entitled 'The Job Tables'."
3819      INDEX  { jmGeneralJobSetIndex, jmJobIndex }
3820      ::= { jmJobTable 1 }
3821
3822  JmJobEntry ::= SEQUENCE {
3823      jmJobIndex                          Integer32 (1..2147483647),
3824      jmJobState                          JmJobStateTC,
3825      jmJobStateReasons1                  JmJobStateReasons1TC,
3826      jmNumberOfInterveningJobs           Integer32 (-2..2147483647),
3827      jmJobKOctetsPerCopyRequested        Integer32 (-2..2147483647),
3828      jmJobKOctetsProcessed               Integer32 (-2..2147483647),
3829      jmJobImpressionsPerCopyRequested    Integer32 (-2..2147483647),
3830      jmJobImpressionsCompleted           Integer32 (-2..2147483647),
3831      jmJobOwner                          JmJobStringTC (SIZE(0..63))
3832  }
3833
```

```
jmJobIndex OBJECT-TYPE
    SYNTAX      Integer32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The sequential, monatonically increasing identifier index for
        the job generated by the server or device when that server or
        device accepted the job.  This index value permits the
        management application to access the other tables to obtain the
        job-specific row entries."
    REFERENCE
        "See Section 3.2 entitled 'The Job Tables and the Oldest Active
        and Newest Active Indexes'.
        See Section 3.5 entitled 'Job Identification'.
        See also

        jmGeneralNewestActiveJobIndex for the largest value of
        jmJobIndex.
        See JmJobSubmissionIDTypeTC for a limit on the size of this
        index if the agent represents it as an 8-digit decimal number."
    ::= { jmJobEntry 1 }



jmJobState OBJECT-TYPE
    SYNTAX      JmJobStateTC
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The current state of the job (pending, processing, completed,
        etc.).  Agents SHALL implement only those states which are
        appropriate for the particular implementation.  However,
        management applications SHALL be prepared to receive all the
        standard job states.

        The final value for this object SHALL be one of: completed,
        canceled, or aborted.  The minimum length of time that the
        agent SHALL maintain MIB data for a job in the completed,
        canceled, or aborted state before removing the job data from
        the jmJobIDTable and jmJobTable is specified by the value of
        the jmGeneralJobPersistence object."
    DEFVAL      { unknown }      -- default is unknown
    ::= { jmJobEntry 2 }

```

```
jmJobStateReasons1 OBJECT-TYPE
    SYNTAX      JmJobStateReasons1TC
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Additional information about the job's current state, i.e.,
        information that augments the value of the job's jmJobState
        object.

        Implementation of any reason values is OPTIONAL, but an agent
        SHOULD return any reason information available.  These values
        MAY be used with any job state or states for which the reason
        makes sense.  Since the Job State Reasons will be more dynamic
        than the Job State, it is recommended that a job monitoring
        application read this object every time jmJobState is read.
        When the agent cannot provide a reason for the current state of
        the job, the value of the jmJobStateReasons1 object and
        jobStateReasonsN attributes SHALL be 0."
    REFERENCE
        "The jobStateReasonsN (N=2..4) attributes provide further
        additional information about the job's current state."
    DEFVAL      { 0 }      -- no reasons
    ::= { jmJobEntry 3 }



jmNumberOfInterveningJobs OBJECT-TYPE
    SYNTAX      Integer32 (-2..2147483647)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of jobs that are expected to complete processing
        before this job has completed processing 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
        attribute when the job is the next job to complete processing
        (or has completed processing)."
    DEFVAL      { 0 }      -- default is no intervening jobs.
    ::= { jmJobEntry 4 }

```

```
jmJobKOctetsPerCopyRequested OBJECT-TYPE
    SYNTAX      Integer32 (-2..2147483647)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The total size in K (1024) octets of the document(s) being
        requested to be processed in the job.  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.

        In computing this value, the server/device SHALL not include
        the multiplicative factors contributed by (1) the number of
        document copies, and (2) the number of job copies, 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 indicates the size of the document(s)
        measured in K octets independent of the number of copies."
    DEFVAL      { -2 }      -- the default is unknown(-2)
    ::= { jmJobEntry 5 }


jmJobKOctetsProcessed OBJECT-TYPE
    SYNTAX      Integer32 (-2..2147483647)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The total number of octets processed by the server or device
        measured in units of K (1024) octets so far.  The agent SHALL
        round the actual number of octets processed up to the next
        higher K.  Thus 0 octets SHALL be represented as '0', 1-1024
        octets SHALL be represented as '1', 1025-2048 octets SHALL be
        '2', etc.  For printing devices, this value is the number
        interpreted by the page description language interpreter rather
        than what has been marked on media.

        For implementations where multiple copies are produced by the
        interpreter with only a single pass over the data, the final
        value SHALL be equal to the value of the
        jmJobKOctetsPerCopyRequested object.  For implementations where
        multiple copies are produced by the interpreter by processing
        the data for each copy, the final value SHALL be a multiple of
        the value of the jmJobKOctetsPerCopyRequested object.

        NOTE - See the impressionsCompletedCurrentCopy and
        pagesCompletedCurrentCopy attributes for attributes that are
        reset on each document copy.
```

```
3970          NOTE - The jmJobKOctetsProcessed object can be used with the
3971          jmJobKOctetsPerCopyRequested object to provide an indication of
3972          the relative progress of the job, provided that the
3973          multiplicative factor is taken into account for some
3974          implementations of multiple copies."
3975      DEFVAL      { 0 }      -- default is no octets processed.
3976      ::= { jmJobEntry 6 }
3977
3978
3979  jmJobImpressionsPerCopyRequested OBJECT-TYPE
3980      SYNTAX      Integer32 (-2..2147483647)
3981      MAX-ACCESS  read-only
3982      STATUS      current
3983      DESCRIPTION
3984          "The total size in number of impressions of the document(s)
3985          submitted.
3986
3987          In computing this value, the server/device SHALL not include
3988          the multiplicative factors contributed by (1) the number of
3989          document copies, and (2) the number of job copies, independent
3990          of whether the device can process multiple copies of the job or
3991          document without making multiple passes over the job or
3992          document data and independent of whether the output is collated
3993          or not.  Thus the server/device computation is independent of
3994          the implementation and reflects the size of the document(s)
3995          measured in impressions independent of the number of copies."
3996      REFERENCE
3997          "See the definition of the term 'impression' in Section 2."
3998      DEFVAL      { -2 }      -- default is unknown(-2)
3999      ::= { jmJobEntry 7 }
4000
4001
4002  jmJobImpressionsCompleted OBJECT-TYPE
4003      SYNTAX      Integer32 (-2..2147483647)
4004      MAX-ACCESS  read-only
4005      STATUS      current
4006      DESCRIPTION
4007          "The total number of impressions completed for this job so far.
4008          For printing devices, the impressions completed includes
4009          interpreting, marking, and stacking the output.  For other
4010          types of job services, the number of impressions completed
4011          includes the number of impressions processed.
4012
4013          NOTE - See the impressionsCompletedCurrentCopy and
4014          pagesCompletedCurrentCopy attributes for attributes that are
4015          reset on each document copy.
4016
4017          NOTE - The jmJobImpressionsCompleted object can be used with
4018          the jmJobImpressionsPerCopyRequested object to provide an
4019          indication of the relative progress of the job, provided that
4020          the multiplicative factor is taken into account for some
4021          implementations of multiple copies."
```

```
4022        REFERENCE
4023            "See the definition of the term 'impression' in Section 2 and
4024            the counting example in Section 3.4 entitled 'Monitoring Job
4025            Progress'."
4026        DEFVAL      { 0 }      -- default is no octets
4027        ::= { jmJobEntry 8 }
4028
4029
4030
4031    jmJobOwner OBJECT-TYPE
4032        SYNTAX      JmJobStringTC (SIZE(0..63))
4033        MAX-ACCESS  read-only
4034        STATUS      current
4035        DESCRIPTION
4036            "The coded character set name of the user that submitted the
4037            job.  The method of assigning this user name will be system
4038            and/or site specific but the method MUST insure that the name
4039            is unique to the network that is visible to the client and
4040            target device.
4041
4042            This value SHOULD be the most authenticated name of the user
4043            submitting the job."
4044        REFERENCE
4045            "See the OBJECT compliance macro for the minimum maximum length
4046            required for conformance."
4047        DEFVAL      { ''H }      -- empty string
4048        ::= { jmJobEntry 9 }
4049
4050
4051
4052
```

```
-- The Attribute Group (MANDATORY)

-- The jmAttributeGroup consists entirely of the jmAttributeTable.
--
-- Implementation of the objects in this group is MANDATORY.
-- See Section 3.1 entitled 'Conformance Considerations'.
-- An agent SHALL implement any attribute if (1) the server or device
-- supports the functionality represented by the attribute and (2) the
-- information is available to the agent.

jmAttribute  OBJECT IDENTIFIER ::= { jobmonMIBObjects 4 }



jmAttributeTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF JmAttributeEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The jmAttributeTable SHALL contain attributes of the job and
        document(s) for each job in a job set.  Instead of allocating
        distinct objects for each attribute, each attribute is
        represented as a separate row in the jmAttributeTable."
    REFERENCE
        "The MANDATORY-GROUP macro specifies that this group is
        MANDATORY.  An agent SHALL implement any attribute if (1) the
        server or device supports the functionality represented by the
        attribute and (2) the information is available to the agent. "
    ::= { jmAttribute 1 }


jmAttributeEntry  OBJECT-TYPE
    SYNTAX      JmAttributeEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Attributes representing information about the job and
        document(s) or resources required and/or consumed.

        Each entry in the jmAttributeTable is a per-job entry with an
        extra index for each type of attribute (jmAttributeTypeIndex)
        that a job can have and an additional index
        (jmAttributeInstanceIndex) for those attributes that can have
        multiple instances per job.  The jmAttributeTypeIndex object
        SHALL contain an enum type that indicates the type of attribute
        (see the JmAttributeTypeTC textual-convention).  The value of
        the attribute SHALL be represented in either the
        jmAttributeValueAsInteger or jmAttributeValueAsOctets objects,
        and/or both, as specified in the JmAttributeTypeTC textual-
        convention.

```

```
4105          The agent SHALL create rows in the jmAttributeTable as the
4106          server or device is able to discover the attributes either from
4107          the job submission protocol itself or from the document PDL.
4108          As the documents are interpreted, the interpreter MAY discover
4109          additional attributes and so the agent adds additional rows to
4110          this table.  As the attributes that represent resources are
4111          actually consumed, the usage counter contained in the
4112          jmAttributeValueAsInteger object is incremented according to
4113          the units indicated in the description of the JmAttributeTypeTC
4114          enum.
4115
4116          The agent SHALL maintain each row in the jmJobTable for at
4117          least the minimum time after a job completes as specified by
4118          the jmGeneralAttributePersistence object.
4119
4120          Zero or more entries SHALL exist in this table for each job in
4121          a job set."
4122      REFERENCE
4123          "See Section 3.3 entitled 'The Attribute Mechanism' for a
4124          description of the jmAttributeTable."
4125      INDEX  { jmGeneralJobSetIndex, jmJobIndex, jmAttributeTypeIndex,
4126      jmAttributeInstanceIndex }
4127      ::= { jmAttributeTable 1 }
4128
4129  JmAttributeEntry ::= SEQUENCE {
4130      jmAttributeTypeIndex                JmAttributeTypeTC,
4131      jmAttributeInstanceIndex            Integer32 (1..32767),
4132      jmAttributeValueAsInteger           Integer32 (-2..2147483647),
4133      jmAttributeValueAsOctets            OCTET STRING(SIZE(0..63))
4134  }
4135
```

```
4136  jmAttributeTypeIndex OBJECT-TYPE
4137      SYNTAX      JmAttributeTypeTC
4138      MAX-ACCESS  not-accessible
4139      STATUS      current
4140      DESCRIPTION
4141          "The type of attribute that this row entry represents.
4142
4143          The type MAY identify information about the job or document(s)
4144          or MAY identify a resource required to process the job before
4145          the job start processing and/or consumed by the job as the job
4146          is processed.
4147
4148          Examples of job attributes (i.e., apply to the job as a whole)
4149          that have only one instance per job include:
4150          jobCopiesRequested(90), documentCopiesRequested(92),
4151          jobCopiesCompleted(91), documentCopiesCompleted(93), while
4152          examples of job attributes that may have more than one instance
4153          per job include:  documentFormatIndex(37), and
4154          documentFormat(38).
4155
4156          Examples of document attributes (one instance per document)
4157          include: fileName(34), and documentName(35).
4158
4159          Examples of required and consumed resource attributes include:
4160          pagesRequested(130), mediumRequested(170), pagesCompleted(131),
4161          and mediumConsumed(171), respectively."
4162      ::= { jmAttributeEntry 1 }
4163
4164
4165
4166  jmAttributeInstanceIndex OBJECT-TYPE
4167      SYNTAX      Integer32 (1..32767)
4168      MAX-ACCESS  not-accessible
4169      STATUS      current
4170      DESCRIPTION
4171          "A running 16-bit index of the attributes of the same type for
4172          each job.  For those attributes with only a single instance per
4173          job, this index value SHALL be 1.  For those attributes that
4174          are a single value per document, the index value SHALL be the
4175          document number, starting with 1 for the first document in the
4176          job.  Jobs with only a single document SHALL use the index
4177          value of 1.  For those attributes that can have multiple values
4178          per job or per document, such as documentFormatIndex(37) or
4179          documentFormat(38), the index SHALL be a running index for the
4180          job as a whole, starting at 1."
4181      ::= { jmAttributeEntry 2 }
4182
```

```
4183  jmAttributeValueAsInteger OBJECT-TYPE
4184      SYNTAX      Integer32 (-2..2147483647)
4185      MAX-ACCESS  read-only
4186      STATUS      current
4187      DESCRIPTION
4188          "The integer value of the attribute.  The value of the
4189          attribute SHALL be represented as an integer if the enum
4190          description in the JmAttributeTypeTC textual-convention
4191          definition has the tag: 'INTEGER:'.
4192
4193          Depending on the enum definition, this object value MAY be an
4194          integer, a counter, an index, or an enum, depending on the
4195          jmAttributeTypeIndex value.  The units of this value are
4196          specified in the enum description.
4197
4198          For those attributes that are accumulating job consumption as
4199          the job is processed as specified in the JmAttributeTypeTC
4200          textual-convention, SHALL contain the final value after the job
4201          completes processing, i.e., this value SHALL indicate the total
4202          usage of this resource made by the job.
4203
4204          A monitoring application is able to copy this value to a
4205          suitable longer term storage for later processing as part of an
4206          accounting system.
4207
4208          Since the agent MAY add attributes representing resources to
4209          this table while the job is waiting to be processed or being
4210          processed, which can be a long time before any of the resources
4211          are actually used, the agent SHALL set the value of the
4212          jmAttributeValueAsInteger object to 0 for resources that the
4213          job has not yet consumed.
4214
4215          Attributes for which the concept of an integer value is
4216          meaningless, such as fileName(34), jobName, and
4217          processingMessage, do not have the 'INTEGER:' tag in the
4218          JmAttributeTypeTC definition and so an agent SHALL always
4219          return a value of '-1' to indicate 'other' for the value of the
4220          jmAttributeValueAsInteger object for these attributes.
4221
4222          For attributes which do have the 'INTEGER:' tag in the
4223          JmAttributeTypeTC definition, if the integer value is not (yet)
4224          known, the agent either (1) SHALL not materialize the row in
4225          the jmAttributeTable until the value is known or (2) SHALL
4226          return a '-2' to represent an 'unknown' counting integer value,
4227          a '0' to represent an 'unknown' index value, and a '2' to
4228          represent an 'unknown(2)' enum value."
4229      DEFVAL      { -2 }      -- default value is unknown(-2)
4230      ::= { jmAttributeEntry 3 }
4231
```

```
4232  jmAttributeValueAsOctets OBJECT-TYPE
4233      SYNTAX      OCTET STRING(SIZE(0..63))
4234      MAX-ACCESS  read-only
4235      STATUS      current
4236      DESCRIPTION
4237          "The octet string value of the attribute.  The value of the
4238          attribute SHALL be represented as an OCTET STRING if the enum
4239          description in the JmAttributeTypeTC textual-convention
4240          definition has the tag: 'OCTETS:'.
4241
4242          Depending on the enum definition, this object value MAY be a
4243          coded character set string (text), such as 'JmUTF8StringTC', or
4244          a binary octet string, such as 'DateAndTime'.
4245
4246          Attributes for which the concept of an octet string value is
4247          meaningless, such as pagesCompleted, do not have the tag
4248          'OCTETS:' in the JmAttributeTypeTC definition and so the agent
4249          SHALL always return a zero length string for the value of the
4250          jmAttributeValueAsOctets object.
4251
4252          For attributes which do have the 'OCTETS:' tag in the
4253          JmAttributeTypeTC definition, if the OCTET STRING value is not
4254          (yet) known, the agent either SHALL not materialize the row in
4255          the jmAttributeTable until the value is known or SHALL return a
4256          zero-length string."
4257      DEFVAL      { ''H }      -- empty string
4258      ::= { jmAttributeEntry 4 }
4259
```

```
-- Notifications and Trapping
-- Reserved for the future

jobmonMIBNotifications  OBJECT IDENTIFIER  ::= { jobmonMIB 2 }



-- Conformance Information

jmMIBConformance OBJECT IDENTIFIER ::= { jobmonMIB 3 }



-- compliance statements
jmMIBCompliance MODULE-COMPLIANCE
    STATUS  current
    DESCRIPTION
        "The compliance statement for agents that implement the
        job monitoring MIB."
    MODULE -- this module
    MANDATORY-GROUPS {
        jmGeneralGroup, jmJobIDGroup, jmJobGroup, jmAttributeGroup }

    OBJECT   jmGeneralJobSetName
    SYNTAX   JmUTF8StringTC (SIZE(0..8))
    DESCRIPTION
        "Only 8 octets maximum string length NEED be supported by the
        agent."

    OBJECT   jmJobOwner
    SYNTAX   JmJobStringTC (SIZE(0..16))
    DESCRIPTION
        "Only 16 octets maximum string length NEED be supported by the
        agent."

-- There are no CONDITIONALLY MANDATORY or OPTIONAL groups.

    ::= { jmMIBConformance 1 }
```

```
4299  jmMIBGroups      OBJECT IDENTIFIER ::= { jmMIBConformance 2 }
4300
4301  jmGeneralGroup OBJECT-GROUP
4302      OBJECTS {
4303          jmGeneralNumberOfActiveJobs,   jmGeneralOldestActiveJobIndex,
4304          jmGeneralNewestActiveJobIndex, jmGeneralJobPersistence,
4305          jmGeneralAttributePersistence, jmGeneralJobSetName}
4306      STATUS  current
4307      DESCRIPTION
4308          "The general group."
4309      ::= { jmMIBGroups 1 }
4310
4311
4312
4313  jmJobIDGroup OBJECT-GROUP
4314      OBJECTS {
4315          jmJobIDJobSetIndex, jmJobIDJobIndex }
4316      STATUS  current
4317      DESCRIPTION
4318          "The job ID group."
4319      ::= { jmMIBGroups 2 }
4320
4321
4322
4323  jmJobGroup OBJECT-GROUP
4324      OBJECTS {
4325          jmJobState, jmJobStateReasons1, jmNumberOfInterveningJobs,
4326          jmJobKOctetsPerCopyRequested, jmJobKOctetsProcessed,
4327          jmJobImpressionsPerCopyRequested, jmJobImpressionsCompleted,
4328          jmJobOwner }
4329      STATUS  current
4330      DESCRIPTION
4331          "The job group."
4332      ::= { jmMIBGroups 3 }
4333
4334
4335
4336  jmAttributeGroup OBJECT-GROUP
4337      OBJECTS {
4338          jmAttributeValueAsInteger, jmAttributeValueAsOctets }
4339      STATUS  current
4340      DESCRIPTION
4341          "The attribute group."
4342      ::= { jmMIBGroups 4 }
4343
4344
4345  END
```

5. Appendix A - Implementing the Job Life Cycle

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

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

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

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

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

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

6. APPENDIX B - Support of Job Submission Protocols

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

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

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

7. References

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

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

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

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

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

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

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

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

[ISO 8859] ISO/IEC 8859-1:1987, "Information technology -- 8-bit single byte coded graphic character sets - Part 1: Latin alphabet No. 1, JTC1/SC2."

[ISO 2022] ISO/IEC 2022:1994 - "Information technology -- Character code structure and extension techniques", JTC1/SC2.

[ISO-3166] ISO 3166:1988 (E/F) - Codes for representation of names of countries - The International Organization for Standardization, 3rd edition, 1988-08-15."

[ISO-10646] ISO/IEC 10646-1:1993, "Information technology -- Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane, JTC1/SC2.

[iso-dpa] ISO/IEC 10175 Document Printing Application (DPA). See ftp://ftp.pwg.org/pub/pwg/dpa/

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

[JIS X0208] JIS X0208-1990, "Japanese two byte coded character set."

[mib-II] MIB-II, RFC 1213.

[print-mib] Smith, R., Wright, F., Hastings, T., Zilles, S. and Gyllenskog, J., "Printer MIB", RFC 1759, proposed IETF standard, March 1995. See also [print-mib-draft].

[print-mib-draft] Turner, R., "Printer MIB", work in progress, on the standards track as a draft standard: <draft-ietf-printmib-mib-info-02.txt>, October 15, 1997.

[protomap] Bergman, R., "Job Submission Protocol Mapping Recommendations for the Job Monitoring MIB," work in progress as an informational RFC. See <draft-bergman-printmib-job-protomap-01.txt>, January 12, 1998.

[pwg] The Printer Working Group is a printer industry consortium open to any individuals. For more information, access the PWG web page: http://www.pwg.org

[req-words] S. Bradner, "Keywords for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

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

[RFC-1766] Avelstrand, H., "Tags for the Identification of Languages", RFC 1766, March 1995.

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

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

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

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

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

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

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

8. Author's Addresses

Ron Bergman
Dataproducts Corp.
1757 Tapo Canyon Road
Simi Valley, CA 93063-3394

Phone: 805-578-4421
Fax: 805-578-4001
Email: rbergman@dpc.com

Tom Hastings
Xerox Corporation, ESAE-231
701 S. Aviation Blvd.
El Segundo, CA 90245

Phone: 310-333-6413
Fax: 310-333-5514
EMail: hastings@cp10.es.xerox.com

Scott A. Isaacson
Novell, Inc.
122 E 1700 S
Provo, UT 84606

Phone: 801-861-7366
Fax: 801-861-4025

EMail: scott_isaacson@novell.com

Harry Lewis
IBM Corporation
6300 Diagonal Hwy
Boulder, CO 80301

Phone: (303) 924-5337
Fax:
Email: harryl@us.ibm.com

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

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

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

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

http://www.pwg.org/

Other Participants:

Chuck Adams - Tektronix
Jeff Barnett - IBM
Keith Carter, IBM Corporation
Jeff Copeland - QMS
Andy Davidson - Tektronix
Roger deBry - IBM
Mabry Dozier - QMS
Lee Ferrel - Canon
Steve Gebert - IBM
Robert Herriot - Sun Microsystems Inc.
Shige Kanemitsu - Kyocera
David Kellerman - Northlake Software
Rick Landau - Digital
Pete Loya - HP
Ray Lutz - Cognisys
Jay Martin - Underscore
Mike MacKay, Novell, Inc.
Stan McConnell - Xerox
Carl-Uno Manros, Xerox, Corp.
Pat Nogay - IBM
Bob Pentecost - HP
Rob Rhoads - Intel

David Roach - Unisys
Stuart Rowley - Kyocera
Hiroyuki Sato - Canon
Bob Setterbo - Adobe
Gail Songer, EFI
Mike Timperman - Lexmark
Randy Turner - Sharp
William Wagner - Digital Products
Jim Walker - Dazel
Chris Wellens - Interworking Labs
Rob Whittle - Novell
Don Wright - Lexmark
Lloyd Young - Lexmark
Atsushi Yuki - Kyocera
Peter Zehler, Xerox, Corp.

9. INDEX

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

printerResolutionUsed	61
printQualityRequested	61
printQualityUsed	61
processingMessage	52
processingMessageNaturalLangTag	53
queueNameRequested	57
serverAssignedJobName	55
sheetCompletedCopyNumber	63
sheetCompletedDocumentNumber	63
sheetsCompleted	66
sheetsCompletedCurrentCopy	66
sheetsRequested	66
sides	61
submittingApplicationName	56
submittingServerName	56
tonerDensityRequested	61
tonerDensityUsed	62
tonerEcomonyRequested	61
tonerEcomonyUsed	61