Table 1 Basic Print Interface Requests and Responses

Operation	Request Parameters	Response Parameters	Note
AddPrintDocument	ElementsNaturalLanguageRequested(optional), Document Ticket (optional) JobId, LastDocument(optional), RequestingUserName, DocumentData	DocumentNumber, UnsupportedElements(optional)	
AddPrintUri	DocumentUri, ElementsNaturalLanguageRequested(optional), Document Ticket (optional) JobId, LastDocument(optional), RequestingUserName	DocumentNumber, UnsupportedElements(optional)	
CancelPrintDocument	DocumentNumber, ElementsNaturalLanguage(optional), JobId, Message (optional) RequestingUserName		
CancelPrintJob	ElementsNaturalLanguage(optional), JobId, Message (optional) RequestingUserName		
CancelMyPrintJobs	JobIds (optional), Message (optional), ElementsNaturalLanguage(optional), RequestingUserName	JobIds (optional)	1
ClosePrintJob	JobId, RequestingUserName		
CreatePrintJob	ElementsNaturalLanguage(optional), Job Ticket (optional) RequestingUserName	JobId, UnsupportedElements(optional)	
GetActivePrintJobs	ElementsNaturalLanguageRequested(optional), Limit(optional) RequestingUserName	ElementsNaturalLanguage(optional) JobSummaries (includes JobID, JobName, JobOriginatingUserName, JobState and perhaps JobStateReasons)(optional)	
GetPrintDocumentElements	Document Number, ElementsNaturalLanguageRequested(optional), JobId, RequestingUserName	DocumentElements(optional), ElementsNaturalLanguage(optional)	
GetPrintDocuments	ElementsNaturalLanguageRequested(optional), JobId, RequestingUserName	Documents(list of DocumentSummaries)(optional), ElementsNaturalLanguage(optional) JobID, JobName	
GetPrintJobElements	ElementsNaturalLanguageRequested(optional), JobId, RequestedElements (JobReceipt, JobStatus, or Job Ticket.)(optional) RequestingUserName	JobElements, ElementsNaturalLanguage(optional)	
GetPrintJob History	ElementsNaturalLanguageRequested(optional), Limit(optional) RequestingUserName	ElementsNaturalLanguage(optional) JobSummaries (includes JobID, JobName, JobOriginatingUserName, JobState and perhaps JobStateReasons	
GetPrintServiceElements	ElementsNaturalLanguageRequested(optional), RequestedElements (Service Capabilities, ServiceConfiguration, ServiceDescription, ServiceStatus or DefaultJob Ticket.)(optional) RequestingUserName	ElementsNaturalLanguage(optional) ServiceElements(optional)	
HoldPrintJob	ElementsNaturalLanguageRequested(optional), JobHoldUntil or JobHoldUntilTime, JobId, Message(optional), RequestingUserName		
ReleasePrintJob	ElementsNaturalLanguageRequested(optional), JobId, Message(optional), RequestingUserName		
ResubmitPrintJob	ElementsNaturalLanguageRequested(optional), JobId, Job Ticket (optional) RequestingUserName	JobId, UnsupportedElements(optional)	

Operation	Request Parameters	Response Parameters	Note
ResumePrintJob	ElementsNaturalLanguageRequested(optional), JobId, Message(optional)RequestingUserName		
SendPrintDocument	ElementsNaturalLanguageRequested(optional), Document Ticket (optional) JobId, LastDocument(optional), RequestingUserName, DocumentData	DocumentNumber, UnsupportedElements(optional)	
SendPrintUri	DocumentUri, ElementsNaturalLanguageRequested(optional), Document Ticket (optional) JobId, LastDocument(optional), RequestingUserName	DocumentNumber, UnsupportedElements(optional)	
SuspendCurrentPrintJob	ElementsNaturalLanguage(optional), JobId(optional), Message(optional), RequestingUserName		
ValidatePrintDocument Ticket	ElementsNaturalLanguageRequested(optional), Document Ticket RequestingUserName	UnsupportedElements(optional)	
Validate PrintJob Ticket	ElementsNaturalLanguage(optional), Job Ticket, RequestingUserName	UnsupportedElements(optional)	
SetPrintDocumentElements	DocumentNumber, ElementsNaturalLanguage(optional), SocumentTicket, JobId, Message(optional), RequestingUserName	UnsupportedElements(optional)	
SetPrintJobElements	ElementsNaturalLanguage(optional), Job Ticket, JobId, Message(optional), RequestingUserName	UnsupportedElements(optional)	

1.1.1 Basic Service Operations

The common Basic operations are listed in Table 1; they are concerned with creating and controlling Jobs and Documents within Jobs. The Operations include those by which a client gets Service Elements to allow selection of Services and formulation of Job Tickets. Some of these operations do affect the state of a Job. However, none of these operations directly affect the state or configuration of the Service except to the extent that creating or canceling a Job may initiate a sequence that affects the Service.

1.1.1.1 AddPrintDocument

1.1.1.2 AddPrintURI

1.1.1.3 CancelPrintDocument

The **CancelPrintDocument** operation allows a client to cancel a specified Document in a specified Job of the Cloud Print Service at any time from when the time the Document is created up to, but not including, the time that the Document is Completed, Canceled or Aborted. Because a Document might already be in Processing by the time a CancelPrintDocument request is received, some portion of the Document processing might be completed before the Document is actually canceled.

The **CancelPrintDocumen**t operation does not remove the Document from the Job or the Service, but does set the specified Document's Document State Document Status Element to Canceled and the Document's Document State Reasons Element to an appropriate value. If the Job containing the Document is again submitted using ResubmitPrintJob, the canceled Document is also submitted for processing. Thus CancelPrintDocument has the same semantics as CancelPrintJob which cancels only the processing of the Job but does not delete the Job object itself.

The **CancelPrintDocument** operation does not affect the states of any of the other Documents in the Job. If the Job is in the Processing state and there are more Documents to be processed, the Service does continue to process the un-canceled Documents. If there are no further Documents to process, the Job is advanced to the Completed state.

The Cloud Print Service MUST reject the operation and return an appropriate response message if the operation requestor is not either the Job owner or a Service or System operator or administrator. Otherwise the Service MUST accept or reject the **CancelPrintDocument** request based on the Document's current state and, if the request is accepted, the Service MUST transition the Document to the indicated new state as follows:

Once a “success” response has been sent, the implementation guarantees that the Document will eventually end up in the Canceled state. Between the time that the **CancelPrintDocument** request is accepted and when the Document enters the Canceled Document-state, the Document State Reasons Element MUST contain a value which indicates to any later query that, although the Document might still be Processing, it will eventually end up in the Canceled state.

1.1.1.4 CancelPrintJob

The **CancelPrintJob** operation changes the state of the identified Job to Canceled, provided that the Job is not already in or in a mode leading directly to a termination state. (i.e., Completed, Canceled, or Aborted.) Because a Job might already be active by the time a CancelPrintJob is received, a portion of the Job may be done before the Job is actually terminated.

The Cloud Print Service MUST reject the operation and return an appropriate response message if the operation requestor is not either the Job owner or a Service or System operator or administrator. Otherwise the Service MUST accept or reject the request based on the Job's current state. If the request is accepted, the Job state is transitioned to Canceled and the Service will issue a success response. If the implementation requires some significant time to cancel a Job in the Processing or ProcessingStopped states, the Service MUST set the Job's JobStateReasons to a value indicating that the Job is transitioning to a Canceled state. If the Job already has a JobStateReasons indicating that it is transitioning to a Canceled state, then the Service MUST reject a CancelPrintJob operation

1.1.1.5 CancelCurrentPrintJob

The CancelCurrentPrintJob operation allows a client to cause the Service to terminate processing on the currently processing Job and to move that Job to the Canceled state. As with any other Basic operation directly affecting a Job, this operation is accepted by the Service only if the originator is the Owner of the affected Job(s) or is an Administrator or Operator.

There is the potential that the current Job may have changed between the time a client requests this operation and the time the Service implements it. Therefore, if the intent is to cancel a particular Job the Client MAY include an optional JobId parameter in the request.

1. If the JobId is included in the request and that Job is currently in the Processing or ProcessingStopped state and the operation requestor has access rights to that Job, the Service MUST accept the request and cancel the Job.

2. If no JobId is included in the request and the operation requestor has access rights to the Job currently in the Processing or ProcessingStopped state, the Service MUST accept the request and cancel that Job.
3. If more than one Job is in the Processing or ProcessingStopped state, all currently processing Jobs to which the request originator has access MUST be canceled unless the operation included the optional JobId, in which case only the identified Job is canceled.
4. If the JobId is included in the request and that Job is not currently in the Processing or ProcessingStopped state; or if the requestor does not have access rights to the identified Job, the Service MUST reject the request and return the appropriate error code.
5. If there is no Job currently in the Processing or ProcessingStopped state or if the requestor does not have access rights to any Job that is in the Processing or ProcessingStopped state, the Service MUST reject the request and return the appropriate error code.

1.1.1.6 CancelMyPrintJobs

The CancelMyPrintJobs operation permits a user to cancel all of their own identified non-Terminated Jobs or, if no specific Jobs are identified in the request, to cancel all of their own non-Terminated Jobs in the Service. This operation works like the Cancel-Job operation except that the operation can apply to multiple Jobs. The client specifies the set of candidate Jobs to be canceled by supplying and/or omitting the JobIds. The Service MUST check the access rights of the requesting user against *all* of the candidate Jobs. If *any* of the candidate Jobs are not owned by the requesting user, the Service MUST NOT cancel any Jobs and MUST return the appropriate error status code along with the list of any JobIds that were specifically identified in the operation request but to which the User is not authorized access.

If this check succeeds, then (and only then) the Service MUST accept or reject the request based on the current state of each of the candidate Jobs and must transition each Job to the indicated new state as shown for the antecedent Cancel-My-Jobs operation in the Standard for Internet Printing Protocol (IPP): Job and Printer Extensions Set 2 [PWG5100.11]. If any of the candidate Jobs that were not already in a Terminating state cannot be canceled, the Service MUST NOT cancel any Jobs and MUST return the appropriate error status code along with the list of JobIds for those Jobs which were specifically identified in the operation request but could not be canceled. If the requested Jobs include some Jobs that are already in a terminating state, this circumstance in itself MUST NOT interfere with the canceling of non-terminated candidate Jobs, but SHOULD result in the return of a warning message identifying the specifically identified Jobs that already were in a Terminating state.

1.1.1.7 ClosePrintJob

The ClosePrintJob operation allows a client to close Job inputs to those Services accepting Documents, even when the last Document input operation for the Job (SendPrintDocument, SendPrintURI or AddPrintDocument) did not include the LastDocument Element with a 'true' value. This ClosePrintJob operation supersedes and, if supported by the Service, is preferable to the practice of using a SendPrintDocument with no Document Data but with a LastDocument Element containing a 'true' value to close inputs.

The Service MUST reject this operation request if the target Job is not found or if the requestor is not the Job Owner or an Administrator. Otherwise, the Service MUST accept this operation request even if the target Job is already closed and regardless of JobState. Closing the Job MUST cause the Service to reject any subsequent Document input operation for the target Job, but MUST NOT affect the execution of any previously accepted Document input operation.

1.1.1.8 CreatePrintJob

The CreatePrintJob operation allows a Client to request creation of a Job in the Service. Upon creation, the Job is in Pending state and available for scheduling unless a Job Processing instruction prevents this. (e.g., JobHoldUntil puts it in PendingHeld state) The CreatePrintJob operation MUST fail if the Service's IsAcceptingJobs Element value is 'false'.

Job Processing is done on one or more Documents. Unlike the antecedent IPP Print-Job operation, CreatePrintJob may involve more than one Document. In either case, the source(s) of the input Document(s) as well as the destination(s) of the output Document(s) are identified in the Job Ticket information submitted in the CreatePrintJob Request,

Once a Job is created, Documents may be input as part of that Job by SendPrintDocument or SendPrintURI operations. In Service implementations that do not accept multiple Documents (i.e., MultipleDocumentJobsSupported = False), Document input is closed after one Document is accepted. In Service implementations that do accept multiple Documents (i.e., Multiple Document Jobs Supported = True), there may be multiple SendPrintDocument or SendPrintURI operations. There are two methods of indicating when all Documents have been sent:

1. issuing a ClosePrintDocument request
2. issuing a SendPrintDocument, SendPrintURI or AddPrintDocument request with the LastDocument Element = True

To avoid a possible hang condition, Service implementations supporting multiple Document Jobs must also support the Multiple Operation Time Out Element that indicates the minimum number of seconds the Service will wait for the next Send or Add operation before taking some recovery action. If, for some reason, there is a longer period between CreatePrintJob and valid Send or Add operations, or between sequential Send or Add operations, the Client MUST send Send or Add requests, even if they are empty, to reset the timeout. If there is a multiple operation timeout, the Service will take remedial action according to the value that Service has indicated in its Multiple Operation Timeout Action Element.

1.1.1.9 GetPrintDocumentElements

The GetPrintDocumentElements operation allows a Client to obtain detailed information about the specified Document within the specified Job. This operation is parallel to the GetPrintJob-Elements operation, but with the target and response Elements relating to a Document rather than a Job.

The Client requests specific groups of Elements (complex Elements) contained within the Document. The Document Data is not part of the Document and cannot be retrieved using this operation. However the location of the Document Data is available. The allowed values for Requested Elements are Document Receipt, Document Status and Document Ticket. Vendors may extend the allowed values.

The Service MUST return the Document Description Element values that a client supplied in the Document Creation operation (CreatePrintJob, SendPrintDocument or SendPrintURI) or provided in SetPrintDocumentElements operation a plus any additional Document Description Elements that the Service has generated, such as Document State. The Service MUST NOT return any Job level Elements that the Document inherits from the Job level but MUST return Document Elements specified at the Document level. It is NOT REQUIRED that a specific Document include all Elements belonging to a group (since some Elements are optional).

However, it is REQUIRED that the Service support all these group names for the Document object.

1.1.1.10 GetPrintDocuments

The GetPrintDocuments operation allows a client to retrieve the list of Documents belonging to the identified Job. A Document summary containing a group of Document Element names with their values will be returned for each Document in the Job.

This operation is similar to the GetPrint and GetPrint operations except that it returns Elements from Documents rather than identified Jobs. As with the GetPrintDocumentElements operation, the Service MUST return only those Elements that are in the Document Ticket.

1.1.1.11 GetPrintJobElements

The GetPrintJobElements operation allows a Client to obtain detailed information on the specified Job. Unlike the antecedent IPP Get-Job-Attributes operation, the GetPrintJobElements request may not specify individual Elements. Rather, the Client requests specific groups of Elements contained within the Job. The allowed values for RequestedElements are Job Receipt, Job Status, or Job Ticket. Vendors may extend the allowed values.

The Service MUST reject this request if the requestor is not authorized access to the identified Job,

1.1.1.12 GetPrintJobs

The GetPrintJobs operation provides summary information on all Jobs that have reached a terminating state (i.e., Completed, Canceled Aborted). As such, it is similar to the antecedent Get-Jobs operation with the which-Jobs Element set to ‘completed’. Unlike Get-Jobs, GetPrintJobs may not include a Requested Elements argument; rather, it always returns a Job Summary for each terminated Job including JobId, JobName, JobOriginatingUserName, JobState and perhaps JobStateReasons and other service specific information.

When the operation is exercised by a User that is not an Administrator, the Job summary may not include all of the summary information, depending upon site security policy.

1.1.1.13 GetPrintServiceElements

The GetPrintServiceElements operation allows a Client to obtain detailed information on the Elements and their values supported by the Service. Unlike the antecedent IPP Get-Printer-Attributes operation, the GetPrintServiceElements request may not specify individual Elements. Rather, the Client requests information on one or more specific group of Elements. The allowed values for Requested Elements are Service Capabilities, Service Configuration, Service Description, Service Status or DefaultJob Ticket. Vendors may extend the allowed values.

Some Services may accept an additional argument in a GetPrintServiceElements request to further filter the response, much as the antecedent IPP Get-Printer-Attributes operation accepted the Document-Format Element. The individual Service specifications identify such arguments if any, their effect and whether support is mandatory.

In addition to the status message, the Service response includes the set of requested Element names and their values for all supported Elements. The response need not contain the requested Element names for any Elements not supported by the Service.

1.1.1.14 GetActivePrintJobs

The GetActivePrintJobs operation provides summary information on all Jobs in the Pending or Processing state. As such, it is equivalent to the antecedent Get-Jobs operation with the which-Jobs Element set to ‘not-completed’. Unlike the antecedent Get-Jobs operation, GetActivePrintJobs may not include a RequestedElements argument; rather, it always returns a JobSummary for each Active Job with the summary including JobId, JobName, JobOriginatingUserName, JobState and perhaps JobStateReasons and other service specific information.

When the operation is exercised by a User that is not an Administrator or Operator, the Job summary may not include all of the summary information, depending upon site security policy.

1.1.1.15 HoldPrintJob

> NOTE: The antecedent Hold-Job operation and the associated Release-Job operation, as defined in IPP/1.1: Model and Semantics [RFC2911], have been deprecated in later IPP specifications in favor of using the antecedent Set-Job-Attributes operation [RFC3380] to set the Hold-Job-Until or Hold-Job-Until-Time attributes.

The HoldPrintJob operation allows a client acting for the Job Owner or an Administrator or Operator to hold a Pending Job in the queue so that it is not eligible for scheduling. The Job transitions as a result of a HoldPrintJob operation depend upon the current Job state, as indicated for the antecedent Hold-Job operation in paragraph 3.3.5 of IPP/1.1: Model and Semantics [RFC2911] The HoldPrintJob request can specify hold until a specific date-time (JobHoldUntilTime) or according to a keyword (JobHoldUntil), where the keyword can specify a period (such as “third-shift”) or be indefinite. A given HoldPrintJob request can specify only one hold condition. In the case of multiple HoldPrintJob requests, the last accepted request overrides the condition imposed by any previous HoldPrintRequest.

The restraint imposed by a HoldPrintJob is removed by a ReleasePrintJob operation directed to the same Job. If a Service implementation supports HoldPrintJob, it must also support ReleasePrintJob and vice-versa.

If the HoldJob operation is supported, then the ReleaseJob operation MUST be supported, and vice-versa. The OPTIONAL JobHoldUntil or JobHoldUntilTime parameter allows a client to specify whether to hold the Job until a specified time, indefinitely or until a specified time period. The Service MUST accept or reject the request based on the Job's current state and transition the Job to the indicated new state as follows. A HoldJob request is rejected when the identified Job is in the Processing or ProcessingStopped states.

1.1.1.16 ReleasePrintJob

> NOTE: The antecedent Release-Job operation and the associated Hold-Job operation, as defined in IPP/1.1: Model and Semantics [RFC2911], have been deprecated in later IPP specifications in favor of using the antecedent Set-Job-Attributes operation [RFC3380] to set Hold-Job-Until or Hold-Job-Until-Time attributes.

The ReleasePrintJob operation allows a client acting for the Job Owner or an Administrator or Operator to release a previously held Job from the PendingHeld state so that it is eligible for scheduling, provided that there is no other reason to keep the Job in the PendingHeld state. That is, the restraint imposed by a HoldPrintJob operation is removed by a ReleasePrintJob operation directed to the same Job. If a Service implementation supports HoldPrintJob, it must also support ReleasePrintJob and vice-versa.

The Job Transitions as a result of a ReleasePrintJob operation depend upon the current Job state, as indicated for the antecedent Release-Job operation in paragraph 3.3.6 of IPP/1.1: Model and Semantics [RFC2911]

1.1.1.17 ResubmitPrintJob

The ResubmitPrintJob operation allows a client acting for the Job Owner or an Administrator or Operator to resubmit a previously completed Job, but with the option of providing new Job Ticket information (other than input Document Data or input Document Data descriptive information.)

The ResubmitPrintJob operation is applicable only to a RetainedJob. A Retained Job is one which remains in the Service after it has been completed or canceled. This may be incidentally or because it is a saved Job, which is a Completed or Canceled Job with a JobSaveDispostion Element value that indicates that the Job, including Document Data if any, should not be deleted or aged-out after the Job is completed.

If a ResubmitPrintJob operation is accepted, the state of the retained Job is not changed; rather, a new Job is created from the identified retained Job and submitted with an implicit **CreateJob** request.

1. If the ResubmitPrintJob request contains a processing Element that was in the retained Job but with a different value, the value supplied in the ResubmitPrintJob operation MUST override the original value (if supported by the Service).
2. If the ResubmitPrintJob request contains a processing Element that was not in the retained Job, the Element with the value supplied with the ResubmitPrintJob operation MUST be applied (if supported by the Service)
3. For any processing Element in the original retained Job the value of which is not changed in the ResubmitPrintJob request, that Element and its value MUST be applied to newly created Job except that a JobSaveDispostion Element value indicating that the Job should be saved, and certain other Service-specific Element values, MUST NOT be copied but are applied to the new Job only if they are in the ResubmitPrintJob request.

The newly created Job is moved to the Pending or PendingHeld Job state with the same Element values as the original saved Job (except for the save Element). If any of the Documents in the saved Job were passed by reference (SendPrintURI or Send>service>URI), the Service MUST re-fetch the data, since the semantics of RestartPrintJob are to repeat all Job processing. The Service MUST assign new JobUri and JobId values to the newly created Job; the JobDescription Elements that accumulate Job progress, such as JobImpressionsCompleted, JobMediaSheetsCompleted, and JobKOctetsProcessed, MUST be an accurate record for the newly created Job.

The Service MUST accept or reject the ResubmitPrintJob Request based on the authority of the requester and the referenced Job's current state. The Requester must either be the Job owner or an operator or administrator of the Service. The target Job must be retained with a Completed or Canceled state.

1.1.1.18 ResumePrintJob

The ResumePrintJob operation allows a client acting for the Job Owner or an Administrator or Operator to resume the identified Job at the point where it was suspended. Provided that no other condition exists that forces the Job to the PendingStopped state, the Service moves the Job from the ProcessingStopped state to the Pending state and removes the JobSuspended value from the Job's StateReasons Element. If the identified Job is not in the ProcessingStopped state with the JobSuspended value in the Job's StateReasons Element, the Service MUST reject the request and return an appropriate status code, since the Job was not suspended.

If a Service supports SuspendPrintJob or SuspendCurrentPrintJob operations, it MUST support the ResumePrintJob operation, and vice-versa.

1.1.1.19 SendPrintDocument

The SendPrintDocument operation allows a client acting for the Job Owner or an Administrator or Operator to input a Digital Document to a Service as part of an already created Job. In response to the CreatePrintJob, the Service will have returned the JobURI and the JobId. For each Document that the client desires to add to this Job, the client issues a SendPrintDocument request which includes the JobId and contains the entire stream of Document Data for one Document.

If the Service supports this operation but does not support multiple Documents per Job, Document input is closed after the first Document is accepted and the Service MUST reject subsequent SendPrintDocument requests associated with the same Job. Similarly, if the Service does support multiple Documents per Job, the Service MUST reject SendPrintDocument requests associated with a given Job after inputs to that Job have been closed either a ClosePrintJob operation or a previous SendPrintDocument with a 'true' value for the LastDocument Element. Note that the Client may send and the Service must accept a SendPrintDocument request with a 'true' value for the LastDocument Element to close input to that Job, even if that request includes no Document data.

See the Create<system>Job description for discussion of issues relating to excessive delay between multiple SendPrintDocument requests.

The Service MUST reject a SendPrintDocument request and send an appropriate message if:

1. The requestor is not the owner of the identified Job, or is not an Administrator or operator
2. The Service has already closed inputs to the identified Job,
3. The Document size, format and/or compression are not supported by the Service, or
4. The Job is not found.

Otherwise, the Service MUST accept the request, MUST close the Job if the LastDocument Element is asserted, MUST add the supplied Document Data (if any) to the identified Job, and MUST respond to the request.

1.1.1.20 SendPrintUri

The SendPrintUri operation allows a client acting for the Job Owner or an Administrator or Operator to input a Digital Document to a Service as part of an already created Job. As such, the SendPrintUri operation is identical to the SendPrintDocument except that a client supplies a URI reference (DocumentUri Element) rather than the Document Data itself. If a Service supports both operations, clients can use both SendPrintUri and SendPrintDocument operations to add new Documents to an existing multi-Document Job.

As with SendPrintDocument, if the Service supports SendPrintUri but does not support multiple Documents per Job, the Service MUST reject subsequent SendPrintUri requests associated with the same Job. Similarly, if the Service does support multiple Documents per Job, the Service MUST reject SendPrintUri requests associated with a given Job after inputs to that Job have been closed. Job inputs can be closed either by a ClosePrintJob operation or a SendPrintDocument (NOT a SendPrintUri) request with a 'true' value for the LastDocument Element. Note that the Client may send and the Service must accept a SendPrintDocument request with a 'true' value for the LastDocument Element to close input to that Job even if that request includes no Document data.

The Service MUST reject this request and send an appropriate message if:

1. The requestor is not the owner of the identified Job, or is not an Administrator or operator
2. The Service has already closed inputs to the identified Job,
3. The Job is not found
4. The Document size, format and/or compression are not supported by the Service, or
5. The Service does not support the URI Scheme specified.

Otherwise, the Service MUST accept the request, MUST close the Job if the LastDocument Element is asserted, MUST add the Document Data (if any) to the identified Job, and MUST respond to the request. See the Create<system>Job description for discussion of issues relating to excessive delay between multiple SendPrintUri requests.

1.1.1.21 SetPrintDocumentElements

The SetPrintDocumentElements operation allows a Client, operating for the Job Owner or an Administrator, to set the values of identified Elements of the specified Document within the specified Job. This operation is parallel to the SetPrintJobElements and SetPrintServiceElements operations and it follows the same rules for validation, but with the target and response Elements relating to a Document rather than a Job or the Service.

The Client must fully identify the Elements to be set as well as the set values. The only settable Elements are those within the Document Ticket. The Document Data is not part of the Document and cannot be changed using this operation. If a Document was originally submitted without a given settable Element that the SetPrintDocumentElements request attempts to set, the Service adds the specified Element to the Document.

If the client identifies a Document Element but does not specify a value for that Element, then the Service MUST remove the Element and all of its values from the Document. The semantic effect of the client supplying the Element with no value in a SetPrintDocumentElements operation MUST be the same as if the Element had not originally been supplied with the Document. This corresponds to the action of the out-of-band value "DeleteElement" in the antecedent IPP Set-Document-Attributes operation. Any subsequent GetPrintDocumentElements or GetPrintDocuments request MUST NOT return any Element that has been deleted. However, a client can re-establish such a deleted Document Element with any supported value(s) using a subsequent SetPrintDocumentElements operation.

If the client supplies an Element in a SetPrintDocumentElements request with no value and that Element is not present in the Document object, the Service ignores that supplied Element in the request, does not return the Element in the Unsupported Elements group, and returns the ‘success’ status code, provided that there are no other problems with the request.

The validation of the SetPrintDocumentElements request is performed by the Service as if the Document had been submitted originally with the new Element values (and the deleted Elements removed); i.e., all modified Document Elements and values must be supported in combination with the Document Elements not modified. If such a Document Creation operation would have been accepted, then the SetPrintDocumentElements MUST be accepted. If such a Document Creation operation would have been rejected, then the SetPrintDocumentElements MUST be rejected and the Document MUST be unchanged. In addition, if any of the supplied Elements are not supported, are not settable, or the values are not supported, the Service MUST reject the entire operation; the Service MUST NOT set just some of the supplied Elements. That is, SetPrintDocumentElements MUST be implemented as an atomic operation; after the operation, all the supplied Elements MUST be set or all of them MUST NOT be set.

The value of JobMandatoryElements supplied in the original CreatePrintJob request, if any, MUST have no effect on the behavior of the SetPrintDocumentElements operation. Rather, the Service must consider that any Element or Element value in a SetPrintDocumentElements operation is mandatory. The Service MUST reject any request to set a Document Element to an unsupported value or to a value that would conflict with another Document Element value.

The Service MUST respond to the SetPrintDocumentElements operation as defined for the antecedent Set-Document-Attributes operation in the Standard for IPP Document Objects [PWG5100.5]. Although the Document's current state affects whether the Service accepts or rejects the SetPrintDocumentElements request, the operation MUST NOT change the state of the Document object (since the Document is a passive object and the Document state is a subset of the JobState). For example, if the operation creates a request for unavailable resources, the Job (but not the Document) transitions to a new state.

1.1.1.22 SetPrintJobElements

The SetPrintJobElements operation allows a Client operating for the Job Owner or an Administrator, to set the values of identified Elements of the specified Job. The Client must fully identify the Elements to be set as well as the set values. In the response, the Service returns success or rejects the entire request with indications of which Element or Elements could not be set to the specified values.

This operation is parallel to the SetPrintDocumentElements and SetPrintServiceElements operations and it follows the same rules for validation, but with the target and response Elements relating to a Job rather than a Document or the Service

If the client identifies a Job Element but does not specify a value for that Element,, then the Service MUST remove the Element and all of its values from the Job. The semantic effect of the client supplying the Element with no value in a SetPrintJobElements operation MUST be the same as if the Element had not originally been supplied with the Job. This corresponds to the action of the out-of-band value “DeleteElement” in the antecedent IPP Set-Job-Attributes operation. Any subsequent GetPrintJobElements or GetPrintJobs request MUST NOT return any Element that has been deleted. However, a client can re-establish such a deleted Job Element with any supported value(s) using a subsequent SetPrintJobElements operation.

If the client supplies an Element in a SetPrintJobElements request with the DeleteElement value and that Element is not present on the Job object, the Service ignores that supplied Element in the request, does not return the Element in the Unsupported Elements group, and returns the ‘success’ status code, provided that there are no other problems with the request.

The validation of the SetPrintJobElements request is performed by the Service as if the Job had been submitted originally with the new Element values (and the deleted Elements removed); i.e., all modified Job Elements and values must be supported in combination with the Job Elements not modified. If such a Job Creation operation would have been accepted, then the SetPrintJobElements request MUST be accepted. If such a Creation operation would have been rejected, then the SetPrintJobElements MUST be rejected and the Job MUST be unchanged. In addition, if any of the supplied Elements are not supported, are not settable, or the values are not supported, the Service MUST reject the entire operation; the Service MUST NOT partially set some of the supplied Elements. In other words, after the operation, all the supplied Elements MUST be set or none of them MUST be set, thus making the SetPrintJobElements an atomic operation.

The value of JobMandatoryElements supplied in the original CreatePrintJob request, if any, MUST have no effect on the behavior of the SetPrintJobElements operation. Rather, the Service must consider that any Element or Element value in a SetPrintJobElements operation is mandatory. The Service MUST reject any request to set a Job Element to an unsupported value or to a value that would conflict with another Job Element value.

The Service MUST accept or reject the SetPrintJobElements operation according to the rules defined for the antecedent Set-Job-Attributes operation in Internet Printing Protocol (IPP):Job and Printer Set Operations [RFC3380].

1.1.1.23 SuspendCurrentPrintJob

The SuspendCurrentPrintJob operation allows a Client operating for the Job Owner or an Administrator, to suspend a Job by setting a condition in a Job that is currently in the Processing or ProcessingStopped state. This condition, reflected by the JobSuspended value in that Job’s JobStateReasons Element, causes that Job to be in the ProcessingStopped state. The Service is able to processes other Jobs normally, provided that no other inhibiting conditions exist. Note that a Job may be ProcessingStopped state for other reasons and that, once it has been suspended, the Job will remain in the ProcessingStopped state even after the other conditions have been removed.

There is the potential that the current Job may have changed between the time a client requests this operation and the time the Service implements it. Therefore, if the intent is to suspend a particular Job, the Client can include an optional JobId parameter in the request.

The target Job is:

The Job identified by the JobId, if included in the request
If the JobId is not included in the request, any Jobs in the Processing or ProcessingStopped state to which the requestor has access rights.

The Service MUST reject the request and send an appropriate message if:

a. There is no target Job in the Processing or ProcessingStopped state to which the requestor has access rights.
b. The target Job or all potential target Jobs have already been suspended.

The Service MUST accept the request, cancel any target Job(s) that have not been previously suspended, and return an appropriate message if:

1. The target JobId is included in the request and that Job is currently in the Processing or ProcessingStopped state (but is not suspended), and the requestor has access rights,
2. If no JobId is included and the requestor has access rights to the Job that is currently in the Processing or ProcessingStopped state (but is not suspended), the Service MUST accept the request and suspend that Job.
3. If more than one Job is in the Processing or ProcessingStopped state (but are not suspended), all such Jobs MUST be suspended unless the operation request included the optional JobId, in which case only the identified target Job MUST be suspended.
4. If the JobId is included in the request and that Job is not currently in the Processing or ProcessingStopped state; or if the JobId is not included and there is no Job currently in the Processing or ProcessingStopped state, the Service MUST reject the request and return the appropriate error code.
5. If the JobId is included in the request and that Job has been suspended; or if no JobId is included and is currently in the Processing or ProcessingStopped state, the Service MUST reject the request and return the appropriate error code.

The ResumePrintJob operation causes a suspended Job to be released. If a Service supports SuspendCurrentPrintJob operation, it MUST support the ResumePrintJob operation, and vice-versa.

1.1.2 Administrative Service Operations

Administrative Service operations directly affect the Service as a whole or affect the Jobs of multiple Job Owners. Access is reserved for Administrators or Operators. The MFD Administrative Service Operations are listed in Table 1 and are described below.

Table 1 Administrative Operations

Operation	Request Parameters (Note 2)	Response Parameters (Note 3)	Note
CancelPrintJobs	ElementsNaturalLanguage(optional), JobIds(optional), Message (optional) RequestingUserName	JobIds (optional)	1
DisablePrintService	ElementsNaturalLanguage(optional) Message (optional), RequestingUserName		
EnablePrintService	ElementsNaturalLanguage(optional), Message (optional) RequestingUserName	-	
HoldNewPrintJobs	ElementsNaturalLanguageRequested (optional), JobHoldUntil | JobHoldUntilTime, Message(optional), RequestingUserName		
PausePrintService	ElementsNaturalLanguageRequested (optional), Message(optional), RequestingUserName		
PausePrintServiceAfterCurrentJob	ElementsNaturalLanguageRequested (optional), Message(optional), RequestingUserName		
PromotePrintJob	ElementsNaturalLanguageRequested (optional), JobId, Message(optional), PredecessorJobID(optional), RequestingUserName		
ReleaseNewPrintJobs	ElementsNaturalLanguageRequested (optional), Message(optional), RequestingUserName		
RestartPrintService	ElementsNaturalLanguageRequested (optional), IsAcceptingJobs| IsAcceptingResources (optional), Message(optional), RequestingUserName		
ResumePrintJob	ElementsNaturalLanguageRequested(optional), JobId, Message(optional), RequestingUserName		
ResumePrintService	ElementsNaturalLanguageRequested(optional), Message(optional), RequestingUserName		

Operation	Request Parameters (Note 2)	Response Parameters (Note 3)	Note
SetPrintServiceElements	DefaultJob Ticket(optional), RequestingUserName ElementsNaturalLanguageRequested (optional), Capabilities(optional), CapabilitiesReady(optional), Description(optional), Message(optional),	Unsupported Elements(optional)	
ShutdownPrintService	ElementsNaturalLanguageRequested(optional), Message(optional), RequestingUserName		4

Note 1: CancelPrintJobs response includes identified but un-cancellable Jobs

Note 2: The RequestingUserName, is used by the Service to determine whether the requestor is an Administrator, Operator or the Job Owner and is therefore authorized to make the request. Some implementations may require further authentication of the requestor's identity. If the requestor is not determined to have access, the Service MUST reject the request.

Note 3: All responses must correlate to request and indicate whether request was successful or failed.

Note 4: Forcing Service Shutdown may also force the state of any active Jobs to Aborted.

1.1.2.1 CancelPrintJobs

The CancelPrintJobs operation allows the Operator or Administrator of the Service to cancel all identified non-Terminated Jobs or, if no specific Jobs are identified in the request, to cancel all non-Terminated Jobs in the Service. It differs from the CancelPrintJob operation in that it works on a number of Jobs at once. If, following the legal Job state Transitions in Table, the Service cannot successfully cancel all explicitly or implicitly requested Jobs that are not already in the terminated state it MUST NOT cancel any Jobs but MUST return an error code. In this case, the Service MUST also return the list of JobIds for those Jobs that were explicitly identified in the request but could not be canceled.

The set of candidate Jobs to be canceled is specified by the supplied JobIds. If no JobIds are supplied, it is implicit that all Jobs that are not in a Terminating state are to be canceled. As with all Administrative operations, the Service MUST check the access rights of the requesting user. Provided that the requester has access rights, the Service MUST check the current state of each of the candidate Jobs. If any of the candidate Jobs cannot be canceled, the Service MUST NOT cancel any Jobs and MUST return the indicated error status code along with the list of offending JobId values. If there are no Jobs that cannot be canceled, the Service MUST transition each identified Job to the indicated new state as defined for the antecedent Cancel-Jobs operation in paragraph 6.1 of Standard for Internet Printing Protocol (IPP):9 Job and Printer Extensions Set 2 [PWG5100.11].

1.1.2.2 DisablePrintService

The DisablePrintService operation prevents the Service from creating any new Jobs by negating the IsAcceptingJobs Element. This operation has no effect upon the Service State and the Service is still able to process operations other than CreatePrintJob. All previously created or submitted Jobs and all Jobs currently processing continue unaffected.

If the requestor is determined to have proper access, the Service MUST accept this request and MUST negate the IsAcceptingJobs Element.

The IsAcceptingJobs Element value is reaffirmed by the EnablePrintService operation. If an implementation supports DisablePrintService it must also support EnablePrintService and vice-versa.

1.1.2.3 EnablePrintService

The EnablePrintService operation asserts the IsAcceptingJobs Element to allow the Service to accept new CreatePrintJob requests. The operation has no effect upon the Service State or any other operation requests the Service may receive.

If the requestor is determined to have proper access, the Service MUST accept this request and MUST assert the IsAcceptingJobs Element. The Service MUST then be able to accept and implement CreatePrintJob requests, provided that no other inhibiting condition exists.

If a Service implementation supports the DisablePrintService operation, then it must also support EnablePrintService operation and vice-versa.

1.1.2.4 HoldNewPrintJobs

The HoldNewPrintJobs operation allows a client to prevent any new Jobs from being eligible for scheduling by forcing all newly-created Jobs to the PendingHeld state with a JobHoldUntil or JobHoldUntilTime Job Processing Element added, depending upon the Element supplied with the HoldNewPrintJobs operation request. The operation has the same effect as a HoldPrintJobs operation except that any Jobs in the Pending or Processing state when the HoldNewPrintJobs request is accepted are allowed to go to completion, provided that no other conditions or operations prevent this.

The JobHoldUntil parameter allows a client to specify holding new Jobs indefinitely or until a specified named time period. The JobHoldUntilTime parameter allows a client to hold new Jobs until a specified time. Provided that the requestor is authorized and the operation and requested parameters are supported, a Service MUST accept a HoldNewPrintJobs request and MUST add the supplied 'JobHoldUntil' or JobHoldUntilTime Element to the Jobs. This HoldNewPrintJob condition may be cleared by a ReleaseNewPrintJobs operation.

If the HoldNewJobs operation is supported, then the ReleaseNewPrintJobs operation MUST be supported, and vice-versa

1.1.2.5 PausePrintService

The PausePrintService operation allows a client to send the Service to the Stopped state. In this Service state, the Service MUST NOT advance any Job to Job Processing state. Depending on implementation, the PausePrintService operation MAY also stop the Service from continuing to process any current Job, sending the Job to the ProcessingStopped state. That is, depending upon implementation, any Job that is currently in the Processing state may be sent to the ProcessingStopped state as soon as the implementation permits; or the Job may continue to a termination state as determined by other conditions. The Service MUST still accept CreateJob operations to create new Jobs, provided that there are no other conditions preventing it.

If the PausePrintService operation is supported, then the Resume operation MUST also be supported, and vice-versa.

Service State transitions resulting from a PausePrintService operation are the same as defined for the antecedent Pause-Printer operation in paragraph 3.2.7 of IPP/1.1: Model and Semantics [RFC29110. The PausePrintService action should be done as soon as the possible after the request is accepted. If the implementation will take more than negligible time to stop processing (perhaps to finish processing the current Job), the Service may remain in the ‘Processing’ state but MUST add the 'MovingToPaused' value to the Service’s StateReasons Element. When the Service transitions to the 'Stopped' state, it removes the 'MovingToPaused' value and adds the 'Paused' value to the Service’s StateReasons Element. If the implementation permits the current Job to stop in mid processing, the Service transitions directly to the ‘Stopped’ state with the Service’s StateReasons Element set to the 'Paused' value and the current Job transitions to the 'ProcessingStopped' state with the JobStateReasons Element set to the 'Stopped' value.

For any Jobs in the 'Pending' or 'PendingHeld' state, the ‘Stopped' value of the Jobs' JobStateReasons Element also applies. However, the Service need not update those Jobs' JobStateReasons Element and need only return the 'Stopped' value when those Jobs are queried (so-called lazy evaluation).

Provided that the requestor is authorized, the Service MUST accept the PausePrintService request in any Service state and act as defined for the antecedent Pause-Printer operation in paragraph 3.2.7 of IPP/1.1: Model and Semantics [RFC29110].

1.1.2.6 PausePrintServiceAfterCurrentJob

The PausePrintServiceAfterCurrentJob operation allows a client to stop the Service from processing any Jobs once any Jobs currently in Processing are completed. This operation has no effect on the current Jobs and the Service MUST complete the processing of the current Jobs, provided that no other condition or operations preclude it. The Service MUST still accept **CreateJob** operations to create new Jobs, but MUST not cause any Jobs to enter 'Processing'. If the PausePrintServiceAfterCurrentJob operation is supported, then the ResumePrintService operation MUST also be supported.

Service State transitions resulting from a PausePrintServiceAfterCurrentJob operation are as identified for the antecedent Pause-Printer-After-Current-Job operation in IPP: Job and Printer Operations [RFC3998]. Note that, in implementations where the Service implementation is not able to pause Jobs currently in the Processing state, the response to the PausePrintServiceAfterCurrentJob request and the PausePrintService request are exactly the same.

If the implementation will take more than negligible time to finish processing the current Jobs, the Service will remain in the Processing state and must add the 'MovingToPaused' value to the Service’s StateReasons Element. When the Service transitions to the 'Stopped' state, it removes the 'MovingToPaused' value and adds the 'Paused' value to the Service’s StateReasons Element.

For any Jobs in the 'Pending' or 'PendingHeld' state, their state is unchanged but the JobStateReasons Element must be set to the ‘Stopped' value. However, the Service need not update those Jobs' JobStateReasons Element and only need return the 'Stopped' value when those Jobs are queried (so-called lazy evaluation).

Provided that the requestor is authorized, the Service MUST accept the request in any Service state and MUST transition the Service to the indicated new State as follows before returning the operation response as defined for the antecedent Pause-Printer-After-Current-Job operation in IPP: Job and Printer Operations [RFC3998].

1.1.2.7 PromotePrintJob

The PromotePrintJob operation schedules the identified Job to be processed next, after the currently processing Jobs or, if the request includes the predecessor JobId, immediately after the identified predecessor Job. The PromotePrintJob operation is a combination of the IPP Promote-Job and Schedule-Job-After operations. If the predecessor Job is not specified, it acts in the same way as the antecedent IPP Promote-Job operation. If the predecessor Job is specified, it acts the same way as the antecedent IPP Schedule-Job-After operation.

The identified target Job must be in the 'Pending' state. If the identified target Job is not in the 'Pending' state or if the predecessor Job is identified and it is not in the ‘Pending’, ‘Processing’ or ‘ProcessingStopped’ state, the Service MUST reject the request and return an appropriate status code. If the PromotePrintJob request is accepted, the target Job MUST be processed immediately after the current Jobs or identified predecessor Job reaches a Termination state (Canceled, Completed or Aborted)

Note that the action of this operation is consistent even if a previous PromotePrintJob Request has caused some other Job to be scheduled after the current or predecessor Job; that is, within the rescheduling time limitations of the Service, the Job identified in the last PromotePrintJob Request accepted will be processed next.

1.1.2.8 ReleaseNewPrintJobs

The ReleaseNewPrintJobs operation allows a client to remove the condition initiated by HoldNewPrintJobs and to release all Jobs previously forced to a PendingHeld state by the HoldNewPrintJobs initiated condition so that these Jobs are eligible for scheduling. This is done by removing the 'JobHoldUntilSpecified' and ‘JobHeldByService’ values from the Job's JobStateReasons Element and changing the Jobs’ states to ‘Pending’.

Provided that the requestor is authorized, the Service MUST accept this request in any Service state and the Service MUST remove the 'JobHoldUntilSpecified' value from the Job's JobStateReasons Element for any Job previously forced to a PendingHeld state by the HoldNewPrintJobs initiated condition.

If the ReleaseNewPrintJobs operation is supported, then the HoldNewPrintJobs operation MUST be supported, and vice-versa.

1.1.2.9 RestartPrintService

The RestartPrintService operation causes a Service in any state, even a previously shut down instance of a Service, to be initialized and set to the Idle state, provided that no errors occur or conditions exist that would prevent normal operation. The handling of Jobs that were in the Processing, Pending, PendingHeld, and ProcessingHeld states state prior to Restart is implementation dependent, but a Service Restart MUST be performed as gracefully as possible and in a way preserving the content and integrity of any non-terminated Jobs. Job history data, if supported, SHOULD also be preserved; a particular Service may make this mandatory.

Provided that the requestor is authorized, the Service MUST accept the request RestartPrintService regardless of its current state. Providing that no conditions exist that would normally prevent these actions, the Service MUST reinitialize its State to Idle, clear the StateReasons Element and set the IsAcceptingJobs Element to true.

1.1.2.10 ResumePrintService

The ResumePrintService operation allows a client to cause the Service to resume scheduling Jobs after scheduling has been paused. Provided that the requestor is authorized and the Service supports this operation, a Service MUST accept a ResumePrintService request regardless of the current Service state, corresponding to the actions defined for the antecedent Resume-Printer operation in Internet Printing Protocol/1.1: Model and Semantics [RFC2911]. If there are no other reasons why the Service is in the Stopped state, this operation returns the Service from the Stopped state to the Idle or Processing state from which it was paused, and removes the 'Paused' value to the Service's StateReasons Element.

If the ResumePrintService operation is supported, then the PausePrintService operation MUST be supported, and vice-versa.

1.1.2.11 SetPrintServiceElements

The SetPrintServiceElements operation allows a Client to set the values of identified Elements in the Service, provided that they are settable. Settable Elements may be in Service Capabilities, Service Configuration, Service Description and DefaultJob Ticket but not in Service Status.

The Service MUST reject the entire request with indications of which Element or Elements could not be set if a client request attempts to:

1. Set a non-settable Element (including an Element not in the Service Capabilities, Service Configuration, Service Description or DefaultJob Ticket groups, a read-only Element, and an Element not supported or not supported as a writable Element in the specific Service implementation)
2. Set a settable Element to an invalid value or to a value that conflicts with the values of other Service Elements, including Elements being set in the same request.
3. Set a greater number of Elements in one operation than are supported by the Service implementation (a Service implementation need not support set of more than one Element at a time).

A SetPrintServiceElements operation that specifies an Element but provides no value for that Element is not an error but rather a request to eliminate that Element and whatever value it has.

If there is no reason to reject setting all of the specified Elements to the specified values or elimination of the Element, the Service MUST accept this operation request when it is in the Idle or Stopped state, and SHOULD accept the request when it is in the Processing state.

If the Service accepts the request, only those Elements specified in the request are changed unless the definition of one or more of the set Elements explicitly specifies an effect upon some other Element.

1.1.2.12 ShutdownPrintService

The ShutdownPrintService operation forces the Service to the 'Down' state from any state that it is in, in an orderly manner. That is, the Service MUST stop accepting any further client requests, and MUST stop scheduling Jobs for processing as soon as the implementation allows, although it SHOULD complete the processing of any currently processing Jobs. Once down, the Service will no longer respond to any Client requests other than RestartPrintService request. As with the antecedent IPP Shutdown-Printer operation all Jobs MUST be preserved. As with RestartPrintService, Service shutdown must be performed as gracefully as possible and in a way in preserving the content and integrity of any non-terminated Jobs. Job history data, if supported, SHOULD also be preserved.

Once shut down, a Service can be roused from its Down state by a RestartPrintService operation. If a Service implementation supports ShutdownPrintService it must also support

RestartPrintService and vice-versa. In the down state, the only operation request that a service will respond to is a RestartPrintService operation.

Provided that the requestor is authorized, the Service MUST accept this operation and following an orderly progression, transition to the Down state regardless of the current state of the Service.