JMF ICS

Version 1.3

Date: 7/30/2007

File: ICS-JMF-1.3.doc

Device Messaging / Job Tracking WG

Abstract

This document, the JMF ICS, defines the conformance requirements that relate to communication using JMF. This document specifies two ***Conformance Levels*** of Conformance Requirements. These levels mainly differ in the type of communication between the ***Manager*** and the ***Worker*** and include the conformance requirements for Hot Folders and JMF Messages.

This version applies to interactions using JDF 1.3.

Copyright Notice

Licenses and Trademarks

Table of Contents

Tables

1 Introduction

This ICS builds upon the hot folder based workflow that is provided by the [[Base-ICS]] to add communication using JMF messaging. JMF provides a command and control language that facilitates interaction between Managers and Workers.

This ICS is product sector independent, meaning that it can be used in any workflow where JMF messaging is desired.

Note: in this document, terms identified with ***bold-italics*** are defined in the [[Base-ICS]] if their definition does not appear immediately in this document.

Using [[Base-ICS]] terminology, the [JMF-ICS] specifies the generic parts of the ***Manager Interface*** when it communicates with the ***Worker Interface***. It also specifies the corresponding generic parts of the Worker Interface when it communicates with a Manager Interface.

This ICS includes specifications of JMF Messages used for:

- Determining Supported Messages and submission methods
- Device and Controller discovery
- Dealing with Job submission and return
- General Queue entry handling such as requesting, aborting, holding, suspending, resuming, resubmitting, and removing Queue entries
- Querying and signaling the status of employees, Devices, and Jobs

This ICS defines two conformance levels which are summarized as follows:

Level 1: Defines requirements for:

- Using hot folders for Job submission and return
- Managers to create and manage Persistent Channels

Level 2: Defines requirements for:

- Using JMF messaging for Job submission and return
- Bidirectional Support for a subset of JMF messaging between the Manager and Worker
- Several Messages that MAY be Supported by the Manager and Worker

2 Glossary

This section defines terminology used in this ICS. The terms appear in alphabetic order. If a word is in ***bold-italic***, it is defined in [[Base-ICS]] if it is not defined elsewhere in this section. Elsewhere in ICSs, the first letter of each word of these terms is capitalized.

Table 1: Glossary

Term	Definition
Subscriber	A ***Manager*** or ***Worker*** in a role where it writes a Query or Registration that includes a Subscription Element.
Signaler	A ***Manager*** or ***Worker*** in a role where it consumes a subscription request, i.e. reads and processes a Query or Registration containing a Subscription Element.

3 Conformance Levels

This ICS specifies two ***Conformance Levels*** of ***Conformance Requirements*** – each a subset of [JDF1.3]. These Conformance Requirements are common to all Product-Sector ICSs. These levels differ in the type of communication between the ***Manager*** and the ***Worker***. Table 2 below specifies the Conformance Requirements for Manager Interfaces and Worker.

See Appendix A "*How to Read ICS Documents*" in [Base-ICS] for an explanation of Conformance Tables.

To be conformant to a level of this ICS specified in the first column of Table 2, a Manager MUST conform to the Manager part and a Worker MUST conform to the Worker part of the ICSs and levels specified in all but the first and last columns of Table 2 below.

Table 2: Conformance Levels

Level of this ICS	[Base-ICS]	[JMF-ICS]	[MIS-ICS]	Description
1	2	-	-	Hot Folder Job submission and return. A Worker MUST Support creating Persistent Channels via JMF Elements embedded in **NodeInfo** Resources.
2	2	1	-	JMF used for Job submission, Job return and Queue management. A Manager MUST be able to interpret MIME packages and access their parts. A Worker MUST be able to create MIME packages.

4 Conformance Tables – Resources

The table in this section shows the Conformance Requirements for the **NodeInfo** Resource.

4.1 NodeInfo Resource

Table 3 specifies the conformance requirements for Attributes and Elements for the **NodeInfo** Resource, whether it is linked to a JDF Root Node or a JDF Subnode.

Table 3: NodeInfo

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
TargetRoute	w?	!w		r	r?		Both JMF/QueueSubmissionParams/@ReturnJMF and JMF/QueueSubmissionParams/@ReturnURL MUST be supplied, and they both supersede *TargetRoute*. Level 2 Workers MUST use JMF/QueueSubmissionParams/

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							@ReturnJMF or JMF/ QueueSubmissionParams/@ReturnURL in lieu of *TargetRoute*. **r-Test:** Worker places the JDF instance at the location specified by *TargetRoute* when execution completes or aborts.
JMF	w?			r			This JMF Element is an alternate way for a Manager to start a Persistent Channel. Messages appearing in this JMF Element are limited to Query and Registration Messages with Subscriptions. Command, Response, Signal, and Acknowledge messages MUST NOT appear in this JMF Element. Query or Registration Messages that do not have Subscriptions also MUST NOT appear in this JMF Element. See section 5.4.1Persistent Channels for conformance requirements about creating and closing Persistent Channels. The Worker MUST NOT respond to this JMF with a Response JMF. This JMF Element MUST NOT be used for any Persistent Channels that are not explicitly related to the containing JDF Instance. **r-Test:** Subscriptions defined in this JMF Element are processed, and signals are sent back based on them. See Table 4: JMF.

5 Conformance Tables – JMF Instances

This section contains Conformance Tables that specify conformance requirements for JMF Messages:

5.1 Manager/Worker versus Producer/Consumer

For sections 5.1, 5.3 , and some of the sub-sections in section 5.4, the Conformance Table columns for Manager and Worker are relabeled ***Producer*** and ***Consumer***. When a Manager sends a JMF Message to a Worker, the Manager is the Producer and the Worker is the Consumer. When a Worker sends a JMF Message to a Manager, the Worker is the Producer and the Manager is the Consumer.

5.2 JMF Root Node

This ICS specifies the JMF Messages that are common to all areas where JMF is used.

This ICS also describes the conformance requirements for all of the Messages identified in Table 13: List of JMF Messages.

This ICS describes both the JMF Message formats and the transfer protocol. Managers and Workers MUST be able to use HTTP, and MAY use HTTPS, for the transmission and reception of Messages except Messages that are embedded in **NodeInfo** Resources of JDF Instances. For the remainder of this document, whenever HTTP is referenced, HTTPS MAY also be used.

This section contains the Conformance Table for the JMF Element that is the Root Node of any JMF Message, whether sent by HTTP or as part of a JDF Instance (JDF/ResourcePool/NodeInfo/JMF).

Table 4: JMF

Root Node of: JMF Instance

Referenced by: NodeInfo

Name or Value	*Producer*			*Consumer*			Description
Level ➔	1	2	3	1	2	3	
DeviceID	w?			r			The value identifies the intended recipient of the JMF Element. If the Message is being sent to a Device, this Attribute indicates that Device's *DeviceID*. If the Message is being sent to a Controller, this Attribute identifies that Controller's *ControllerID*. **r-Test:** If the immediate Consumer or any subsequent recipient does not recognize the value of this Attribute, it MUST reject the Message with *ReturnCode* = "121", "Unknown *DeviceID*".
MaxVersion	w			r			
1.3	w			r			A value higher than 1.3 MAY be specified. **r-Test:** Consumer does not write any Elements or Attributes in the response that do not conform to the version specified.
ICSVersions	w←			r?			Has the same semantics as JDF/@ICSVersions. See [[Base-ICS]].
JMF_L1-1.3	w			r?			Specifies that the JMF Element conforms to [JMF-ICS] level 1.
JMF_L2-1.3		w		r?			Specifies that the JMF Element conforms to [JMF-ICS] level 2.
all remaining values	w←			r?			Values specified in other ICSs.
ResponseURL	!w			r?			This ICS does not allow JMF messaging via hot folders.
SenderID	w			r?			Identifies the sender. If the sender is a Device, this Attribute MUST contain the Device's *DeviceID*. If the sender is a Controller, this Attribute MUST contain the Controller's

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
							ControllerID.
TimeStamp	w			r?			Date and time the JMF is sent.
Version	w			r?			
1.3	w			r?			
xmlns	w←			r?			The namespace for JDF may be the default namespace or any prefixed namespace. MUST be present in the JMF Root of a JMF Message, but NEED NOT be present in a JDF Instance (JDF/NodeInfo/JMF).
http://www.CIP4.org/JDFSchema_1_1	w			r?			**Note:** that for all 1.x versions of [JDF1.3], the namespace URI is the same.
xmlns:xsi	w←			r?			MUST be present in the JMF Root of a JMF Message, but NEED NOT be present in a JDF Instance (JDF/ResourcePool/NodeInfo/JMF).
http://www.w3.org/2001/XMLSchema-instance	w			r?			
Message	w			r			Abstract Element(s). See Table 13: List of JMF Messages. See Table 5: Abstract Message.

5.3 JMF Message Families

This section contains Conformance Tables that specify Conformance Requirements for the 6 JMF Message Families.

5.3.1 Abstract Message

Table 5: Abstract Message

Referenced by: JMF

Superclass of: Query, Command, Response, Acknowledge, Signal

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
ID	w			r←			Unique identifier of Message Values MUST be unique for all Messages initiated by the same sender. In other words, the combination of JMF/@SenderID and this *ID* MUST be

Name or Value	Producer			Consumer			Description
Level →	1	2	3	1	2	3	
							unique. If the message is a Query or Command Message, Consumers MUST read *ID*. **r-Test:** The response to the Query or Command specifies a *refID* which matches the *ID* of the Query or Command.
Time	w←			r?			Time at which the Message was generated. This Attribute MUST be specified unless the time is the same as JMF/@TimeStamp.
Type	w			r			**r-Test:** Any actions taken or response generated corresponds to the message type identified by this value.
xsi:type	w			r?			Helps JDF Schema-aware implementations to identify specific Message types.
<Message Family> <value of /JMF/Message/@Type>	w			r?			Example: *CommandSubmitQueueEntry*. • "*Command*" is the Message Family Name, i.e. the actual Element name of the Message. See column 1 of Table 13: List of JMF Messages. • "*SubmitQueueEntry*" is the value of /JMF/Message/@Type. See Table 73: Command – SubmitQueueEntry.

5.3.2 Query

Table 6: Query

Subclass of: Abstract Message

Class for: Query – KnownControllers, Query – KnownDevices, Query – KnownMessages, Query – Occupation, Query – QueueStatus, Query – SubmissionMethods

Name or Value	Producer			Consumer			Description
Level →	1	2	3	1	2	3	
AcknowledgeURL		w←			r?		If a Producer allows the Consumer to send an Acknowledge, the Producer MUST supply this Attribute.
QueryTypeObj	w←			r			Abstract Element that is a placeholder for any descriptive Elements that provide details required for the Query. See [JDF1.3].
Subscription	w←			r			A Consumer MUST Support this Element for

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
							establishing a Persistent Channel. See Table 7: Subscription.

5.3.2.1 Subscription

Also see other ICSs for Subscription contents.

Table 7: Subscription
Referenced by: Query

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
MinDelayTime	w?			r			**r-Test:** Signals related to this Subscription are not sent any more frequently than this interval.
RepeatTime	w?			r			*RepeatTime* MUST NOT be less than *MinDelayTime*. **r-Test:** Signals are generated at the interval specified (+/- 10%).
URL	w			r			**r-Test:** Signals are delivered to the specified URL.
ObservationTarget	w?			r?			See Table 8: ObservationTarget.

5.3.2.2 ObservationTarget

Table 8: ObservationTarget
Referenced by: Subscription

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
ObservationPath	w			r			**r-Test:** When the value specified by the XPath changes a signal is sent.

5.3.3 Response

In Table 9, the Producer is the producer of the Response and the Consumer is the consumer of the original Response. Thus the roles have been exchanged with respect to the original Command or Query tables.

Table 9: Response

Subclass of: Abstract Message

Class for: Response – AbortQueueEntry, Response – HoldQueueEntry, Response – KnownControllers, Response – KnownDevices, Response – KnownMessages, Response – Occupation, Response – QueueStatus, Response – RemoveQueueEntry, Response – RequestQueueEntry, Response – ResubmitQueueEntry, Response – ResumeQueueEntry, Response – ReturnQueueEntry, Response – SetQueueEntryPosition, Response – SetQueueEntryPriority, Response – StopPersistentChannel, Response – SubmissionMethods, Response – SubmitQueueEntry, Response – SuspendQueueEntry

Name or Value	*Producer*			*Consumer*			Description
Level ➔	1	2	3	1	2	3	
Acknowledged		w←			r?		The Producer MUST supply this Attribute with a value of "*true*" when it will send an asynchronous Acknowledge later.
refID		w←			r?		See [JDF1.3].
ReturnCode		w←			r?		If an error occurs, a Producer MUST write a nonzero value. See [JDF1.3] Appendix D for a list of Supported values. A Consumer MUST be able to detect nonzero values.
Subscribed		w←			r?		The Producer MUST supply this Attribute if the Query contained a Subscription (see Table 6: Query).
true		w←			r?		A level 2 Producer MUST accept Subscriptions for Persistent Channels in Queries.

5.3.4 Signal

Table 10: Signal

Subclass of: Abstract Message

Name or Value	*Producer*			*Consumer*			Description
Level ➔	1	2	3	1	2	3	
LastRepeat	w←			r?			If the Producer has closed the Persistent Channel the value of this Attribute MUST be "*true*".

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
refID	w←			r?			If the Signal is a result of a Subscription, the Producer MUST supply the ID of the Subscription Query.
QueryTypeObj	w			r?			If the Signal is the result of a hardwired configuration, the Producer MUST supply a QueryTypeObj that specifies the corresponding query parameters that apply to this Signal. If the Signal is the result of a subscription, the same QueryTypeObj that was in the subscription (or one with additional context added) MUST be supplied here. See [JDF1.3].

5.3.5 Command

Table 11: Command

Subclass of: Abstract Message

Class for: Command – AbortQueueEntry, Command – HoldQueueEntry, Command – RemoveQueueEntry, Command – RequestQueueEntry, Command – ResubmitQueueEntry, Command – ResumeQueueEntry, Command – ReturnQueueEntry, Command – SetQueueEntryPosition, Command – SetQueueEntryPriority, Command – StopPersistentChannel, Command – SubmitQueueEntry, Command – SuspendQueueEntry

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
AcknowledgeURL		w←			r?		If a Producer allows the Consumer to send an Acknowledge, the Producer MUST supply this Attribute.
CommandTypeObj		w←			r		Abstract Element that is a placeholder for any descriptive Elements that provide details required for the Command. See [JDF1.3].

5.3.6 Acknowledge

Table 12: Acknowledge

Subclass of: Abstract Message

Class for: Acknowledge – AbortQueueEntry, Acknowledge – ResumeQueueEntry, Acknowledge – ReturnQueueEntry, Acknowledge – SubmitQueueEntry, Acknowledge – SuspendQueueEntry

Name or Value	*Producer*			*Consumer*			Description
Level ➔	1	2	3	1	2	3	
AcknowledgeType		w←			r?		
all values		w←			r?		
refID		w			r?		ID of the Message being Acknowledged.
ReturnCode		w←			r?		A non-zero return code indicates failure. If an error can occur, a Producer MUST be capable of writing at least one nonzero value. See [JDF1.3] Appendix D for a list of Supported values. A Consumer MUST be able to detect nonzero values. How a Consumer handles nonzero value is implementation dependent.

5.4 JMF Handshaking

5.4.1 Persistent Channels

5.4.1.1 Persistent Channels Created via JMF Embedded in NodeInfo Resources

Workers MUST search the submitted JDF Instance for embedded JMF subscriptions in JDF Nodes that they identify as executable, the parents of the executable JDF Nodes, and the Root Node. JMF subscriptions embedded in other Nodes MAY be ignored by Workers.

5.4.1.2 Creating Persistent Channels

A Manager or Worker sends JMF Signals to another Controller or Device in what is called a Persistent Channel. Persistent Channels are created by sending a Query or Registration Message that includes a Subscription Element.

A Level 1 Manager desiring to establish a Persistent Channel MUST send the Query or Registration Message as a Subelement of the **NodeInfo** Resource(s) that is/are linked as input(s) to JDF Node(s) in the **JDF Instance** (see Table 3: NodeInfo). A Level 1 Worker MUST Support queries and registrations with subscriptions in Messages in linked **NodeInfo** Resources.

A Level 2 ***Subscriber*** MAY send the Query or Registration Message either as a Subelement of the **NodeInfo** Resource or as a separate JMF Query or Registration Message via HTTP. A Level 2 ***Signaler*** MUST Support both Messages with subscriptions in linked **NodeInfo** Resources, as well as Messages with subscriptions sent via HTTP.

5.4.1.3 Closing Persistent Channels

Closing a Persistent Channel means that additional Messages related to the subscription MUST NOT be created. Messages previously created MAY still be pending delivery. ***Subscribers*** MUST be able to receive and process further messages at the URL defined in the original subscription.

The Worker MUST close the Persistent Channel that is set up through the **NodeInfo** Input Resource when execution of the Node that links to the Resource as an input completes or is aborted, or the related QueueEntry/@Status changes to "*PendingReturn*", whichever happens first.

Persistent Channels that relate to a particular Job, Node, or Queue entry that are created with Messages sent via HTTP MUST be closed when the corresponding Job Node completes or aborts or the related QueueEntry/@Status changes to "*PendingReturn*".

If a Persistent Channel does ***not*** relate to a particular Job, Node, or Queue entry, it MUST be closed by sending a StopPersistentChannel Command.

If a Level 2 ***Subscriber*** is capable of sending a Query with Subscription Elements via HTTP, the ***Subscriber*** MUST also be capable of sending a StopPersistentChannel Command. All Level 2 Subscription Consumers MUST Support StopPersistentChannel Commands.

5.4.2 Asynchronous Acknowledges

A Consumer of a JMF Command or Query Message MUST respond by returning a ResponseTypeObj by one of two methods: synchronously or asynchronously. With certain types of command or query Messages, the Producer of the Message MUST Support the use of asynchronous acknowledges by the Consumer of the Message. These Messages can be identified by the following means:

- The Producer of the Message MUST write the *AcknowledgeURL* Attribute in the Command or Query Element.
- The Producer of the Command or Query MUST read and Support the Response Element's *Acknowledged* Attribute with a value of true in the response to the Command or Query.
- The Producer of the Command or Query MUST Support an Acknowledge Message of the same type.

If a Consumer responds synchronously, it MUST:

- Include a ResponseTypeObj in a Response Element,
- Send the JMF Message in the HTTP response channel.

If a Consumer responds asynchronously, it:

- MUST include the Response Element, with the *Acknowledged* Attribute set to true. The Response Element MAY contain a ResponseTypeObj.
- MUST send the JMF Message in the HTTP response channel.
- MAY include a ResponseTypeObj in an Acknowledge Element,
- MUST send one or more Acknowledge Messages as separate JMF Messages. If this ICS requires the response to the command or query being acknowledged to include one or more ResponseTypeObj Elements, the final Acknowledge Message that is sent MUST include all required ResponseTypeObj Elements.

For a message that does not require Support of asynchronous acknowledges, if the Producer does not Support asynchronous responses to the command or query, and has not provided an *AcknowledgeURL*, the Consumer MUST process the message using exactly one of the following methods:

- The message must be fully processed and a full response sent within the timeframe of an http connection.

OR

- A response MUST be returned which indicates *ReturnCode* = 11, which is a new error code that will appear in JDF 1.4 that will be defined as “Synchronous mode not supported for message.”

6 Conformance Tables – JMF Messages

For some Messages, the Conformance-Table columns for Manager and Worker are relabeled ***Producer*** and ***Consumer***. When a Manager sends a JMF Message to a Worker, the Manager is the Producer and the Worker is the Consumer. When a Worker sends a JMF Message to a Manager, the Worker is the Producer and the Manager is the Consumer.

- If the Manager column contains a “write” for a Message, then the Manager is a Producer and the Worker is a Consumer for that Message.

  For example, for a SubmitQueueEntry Command, the Manager that is acting as a Producer sends a Message to a Worker acting as a Consumer.

- If the Manager column contains a “read” for a Message, then the Worker is a Producer and the Manager is a Consumer for that Message.

  For example, for a SubmitQueueEntry Response, the Worker that is acting as a Producer sends a Message to a Manager acting as a Consumer.

Table 13 specifies the Conformance Requirements for pairs of Message Type (/JMF/Message/@Type Attribute) and Message Family.

This ICS does not limit other ICSs from specifying conformance for additional Message Types.

Note: Table 13 provides the mapping between the Manager/Worker and the Producer/Consumer. In the table below, no **r-Test**s are provided. The test for read conformance is that the Manager or Worker meets the requirements of the tables corresponding to each message.

Table 13: List of JMF Messages

Message Type	Family	Manager			Worker			Description
	Level ➔	1	2	3	1	2	3	
AbortQueueEntry	Command		w			r		See Table 14: Command – AbortQueueEntry.
AbortQueueEntry	Response		r			w		See Table 18: Response – AbortQueueEntry.
AbortQueueEntry	Acknowledge		r?			w?		See Table 21: Acknowledge – AbortQueueEntry.
HoldQueueEntry	Command		w?			r?		See Table 22: Command – HoldQueueEntry.
HoldQueueEntry	Response		r?			w?		See Table 23: Response – HoldQueueEntry.
KnownControllers	Query		w? r?			r? w?		Controller/Device Registration Servers SHOULD Support receiving and processing the KnownControllers Query Message. See [JDF1.3] section 5.7.2 for more information. See Table 24: Query –

Message Type	Family	Manager			Worker			Description
	Level →	1	2	3	1	2	3	
								KnownControllers.
KnownControllers	Response		r? w?			w? r?		Controller/Device Registration Servers SHOULD Support producing the KnownControllers Response Message. See [JDF1.3] section 5.7.2 for more information. See Table 25: Response – KnownControllers.
KnownDevices	Query		w r			r w?		See Table 27: Query – KnownDevices.
KnownDevices	Response		r w			w r		See Table 29: Response – KnownDevices.
KnownMessages	Query		w← r			r w?		Managers MUST, and Workers SHOULD, provide the ability for a user/operator to initiate sending a KnownMessages Query. See Table 33: Query – KnownMessages.
KnownMessages	Response		r w			w r		See Table 35: Response – KnownMessages.
Occupation	Query		w? r?			r? w?		See Table 37: Query – Occupation.
Occupation	Response		r? w?			w? r?		See Table 39: Response – Occupation.
QueueStatus	Query		w?			r?		See Table 44: Query – QueueStatus.
QueueStatus	Response		r?			w?		See Table 45: Response – QueueStatus.
RemoveQueueEntry	Command		w			r		See Table 46: Command – RemoveQueueEntry.
RemoveQueueEntry	Response		r			w		See Table 47: Response – RemoveQueueEntry.
RequestQueueEntry	Command		r?			w?		See Table 48: Command – RequestQueueEntry.
RequestQueueEntry	Response		w?			r?		See Table 50: Response – RequestQueueEntry.
ResubmitQueueEntry	Command		w?			r?		See Table 51: Command – ResubmitQueueEntry.
ResubmitQueueEntry	Response		r?			w?		See Table 53: Response –

Message Type	Family	Manager			Worker			Description
	Level →	1	2	3	1	2	3	
								ResubmitQueueEntry.
ResumeQueueEntry	Command		w?			r←		If Worker Supports HoldQueueEntry Message or QueueSubmissionParams/ @Hold, the Worker MUST Support the ResumeQueueEntry Message. See Table 54: Command – ResumeQueueEntry.
ResumeQueueEntry	Response		r?			w←		If Worker Supports HoldQueueEntry Message or QueueSubmissionParams/ @Hold, the Worker MUST Support the ResumeQueueEntry Message. See Table 55: Response – ResumeQueueEntry.
ResumeQueueEntry	Acknowledge		r?			w?		See Table 56: Acknowledge – ResumeQueueEntry.
ReturnQueueEntry	Command		r			w		See Table 57: Command – ReturnQueueEntry.
ReturnQueueEntry	Response		w			r		See Table 59: Response – ReturnQueueEntry.
ReturnQueueEntry	Acknowledge		w?			r?		See Table 60: Acknowledge – ReturnQueueEntry.
SetQueueEntryPosition	Command		w?			r?		See Table 61: Command – SetQueueEntryPosition.
SetQueueEntryPosition	Response		r?			w?		See Table 63: Response – SetQueueEntryPosition.
SetQueueEntryPriority	Command		w?			r←		If Worker Supports QueueSubmissionParams/ @Priority in SubmitQueueEntry command, it must Support the SetQueueEntryPriority Message. See Table 64: Command – SetQueueEntryPriority.
SetQueueEntryPriority	Response		r?			w←		If Worker Supports QueueSubmissionParams/ @Priority in

Message Type	Family	Manager			Worker			Description
	Level →	1	2	3	1	2	3	
								SubmitQueueEntry Command, it must Support the SetQueueEntryPriority Message. See Table 66: Response – SetQueueEntryPriority.
StopPersistentChannel	Command		w←			r		Manager MUST be capable of sending if it Supports a Query with Subscriptions. Worker MUST Support if it accepts a Query with Subscriptions.. See Table 67: Command – StopPersistentChannel.
StopPersistentChannel	Response		r			w		See Table 69: Response – StopPersistentChannel.
SubmissionMethods	Query		w? r			r w?		See Table 70: Query – SubmissionMethods.
SubmissionMethods	Response		rw			wr		See Table 71: Response – SubmissionMethods.
SubmitQueueEntry	Command		w			r		See Table 73: Command – SubmitQueueEntry.
SubmitQueueEntry	Response		r			w		See Table 75: Response – SubmitQueueEntry.
SubmitQueueEntry	Acknowledge		r			w?		See Table 76: Acknowledge – SubmitQueueEntry.
SuspendQueueEntry	Command		w?			r?		See Table 77: Command – SuspendQueueEntry.
SuspendQueueEntry	Response		r?			w?		See Table 78: Response – SuspendQueueEntry.
SuspendQueueEntry	Acknowledge		r?			w?		See Table 79: Acknowledge – SuspendQueueEntry.

6.1 AbortQueueEntry

6.1.1 Command – AbortQueueEntry

Table 14: Command – AbortQueueEntry

Instance of: Command

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
AcknowledgeURL		w←			r?		If a Manager allows the Worker to send an Acknowledge, the Manager MUST supply this Attribute. A Worker MAY respond synchronously if it can abort Queue entries within the time frame of an http connection. If a Worker chooses to acknowledge an AbortQueueEntry, it MUST send an Acknowledge Message once the Queue entry has been aborted.
Type		w			r		
AbortQueueEntry		w			r		
xsi:type		w			r?		
CommandAbortQueueEntry		w			r?		
QueueEntryDef		w			r		See Table 15: QueueEntryDef.
QueueFilter		w?			r		The Manager SHOULD supply this Element. See Table 17: QueueFilter.

6.1.1.1 QueueEntryDef

Table 15: QueueEntryDef

Referenced by: QueueFilter, Command – AbortQueueEntry, Command – HoldQueueEntry, Command – RemoveQueueEntry, Command – ResumeQueueEntry, Command – SuspendQueueEntry

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
QueueEntryID		w			r		**r-Test:** The Queue entry referenced by *QueueEntryID* (and no other) is acted upon (or included in the response for queries) based on the type of the message.

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Comment		w?			r?		See Table 16: Comment.

6.1.1.2 Comment

Table 16: Comment

Referenced by: QueueEntryDef, QueueSubmissionParams

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Name		w			r?		
Instruction		w←			r?		Additional information provided from the Manager to the Worker about the Queue entry being acted upon (e.g. explanation for why it is being aborted).
all remaining values		w?			r?		
<content of Element>		w			r?		

6.1.1.3 QueueFilter

Table 17: QueueFilter

Referenced by: Command – AbortQueueEntry, Command – HoldQueueEntry, Query – QueueStatus, Command – RemoveQueueEntry, Command – ResubmitQueueEntry, Command – ResumeQueueEntry, Command – SetQueueEntryPosition, Command – SetQueueEntryPriority, Command – SubmitQueueEntry, Command – SuspendQueueEntry

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
MaxEntries		w			r		When using QueueFilter with any Message other than QueueStatus, the Manager SHOULD write *MaxEntries* = "0". When using QueueFilter with the QueueStatus Query, the Manager SHOULD write any appropriate value. **r-Test:** The Response to the Message contains no more than the number of Queue entries specified in *MaxEntries*.
QueueEntryDetails		w			r		**r-Test:** The QueueEntry Elements in the

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							Response contain the information corresponding to the value of *QueueEntryDetails*.
None		w←			r		
Brief		w?			r		
all remaining values		!w			r?		
QueueEntryDef		w?			r		The QueueEntryDef Element can be used to filter the response down to a single Queue entry. See Table 15: QueueEntryDef.

6.1.2 Response – AbortQueueEntry

Table 18: Response – AbortQueueEntry

Instance of: Response

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Acknowledged		r			w←		
true		r			w←		
false		r			w←		
Type		r			w		
AbortQueueEntry		r			w		
xsi:type		r?			w		
ResponseAbortQueueEntry		r?			w		
Queue		r			w←		If Response/@Acknowledged=false, Queue MUST be written. Otherwise Queue MAY be written. See Table 19: Queue.

6.1.2.1 Queue

Table 19: Queue

Referenced by: Response – AbortQueueEntry, Acknowledge – AbortQueueEntry, Response – QueueStatus, Response – RemoveQueueEntry, Response – ResubmitQueueEntry, Response – ResumeQueueEntry, Acknowledge – ResumeQueueEntry, Response – SetQueueEntryPosition, Response – SetQueueEntryPriority, Response – SubmitQueueEntry, Acknowledge – SubmitQueueEntry, Response – SuspendQueueEntry, Acknowledge – SuspendQueueEntry

Name or Value	Manager			Worker			Description
Level →	1	2	3	1	2	3	
Status		r?			w		
DeviceID		r?			w		
QueueEntry		r?			w←		See [JDF1.3] 5.6.3.10 for details. See Table 20: QueueEntry.

6.1.2.2 QueueEntry

Table 20: QueueEntry

Referenced by: Queue, Response – SubmitQueueEntry, Acknowledge – SubmitQueueEntry

Name or Value	Manager			Worker			Description
Level →	1	2	3	1	2	3	
JobID		r?			w←		If the QueueEntry is being returned in a Response to a Message other than SubmitQueueEntry, *JobID* MUST be specified.
JobPartID		r?			w←		If the QueueEntry is being returned in a response to a Message other than SubmitQueueEntry, *JobPartID* MUST be specified.
Priority		r?			w←		If the Worker Supports priority of Queue entries, it MUST write this value.
QueueEntryID		r?			w		
Status		r?			w		
SubmissionTime		r?			w←		If the QueueEntry is being written in the Response to the QueueStatus Message, the Worker MUST write the *SubmissionTime*.
Part		r?			w←		If there is an AncestorPool in the Root Node

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							of the submitted JDF and Part Elements appear within it, these Part Elements MUST be copies of AncestorPool/Part of the Root Node of the submitted JDF. See [JDF1.3].

6.1.3 Acknowledge – AbortQueueEntry

Table 21: Acknowledge – AbortQueueEntry

Instance of: Acknowledge

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
AbortQueueEntry		r			w		
xsi:type		r?			w		
AcknowledgeAbortQueueEntry		r?			w		
Queue		r			w←		If the Acknowledge Message is the final Acknowledge Message to the AbortQueueEntry Command, Queue MUST be written. Otherwise Queue MAY be written. See Table 19: Queue.

6.2 HoldQueueEntry

6.2.1 Command – HoldQueueEntry

Table 22: Command – HoldQueueEntry

Instance of: Command

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
HoldQueueEntry		w			r		
xsi:type		w			r?		
CommandHoldQueueEntry		w			r?		
QueueEntryDef		w			r		See Table 15: QueueEntryDef.
QueueFilter		w←			r		It is highly recommended that the Manager supply this Element. See Table 17: QueueFilter.

6.2.2 Response – HoldQueueEntry

Table 23: Response – HoldQueueEntry

Instance of: Response

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
HoldQueueEntry		r			w		
xsi:type		r?			w		
ResponseHoldQueueEntry		r?			w		

6.3 KnownControllers

Note: In this section, the Conformance-Table columns for Manager and Worker are relabeled ***Producer*** and ***Consumer***. See section 5.1 Manager/Worker versus Producer/Consumer.

6.3.1 Query – KnownControllers

Table 24: Query – KnownControllers
Instance of: Query
In: List of JMF Messages

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
KnownControllers		w			r		
xsi:type		w			r?		
QueryKnownControllers		w			r?		

6.3.2 Response – KnownControllers

Note: the Producer in the following tables is returning the response to the Consumers. The Query Producer is the Response Consumer, and the Response Producer is the Query Consumer.

Table 25: Response – KnownControllers
Instance of: Response
In: List of JMF Messages

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
KnownControllers		w			r		
xsi:type		w			r?		
ResponseKnownControllers		w			r?		
JDFController		w			r?		See Table 26: JDFController.

6.3.2.1 JDFController

Table 26: JDFController

Referenced by: Response – KnownControllers

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
ControllerID		w			r?		Identifies the Controller. Each Controller MUST use a fixed *ControllerID*. This value MUST be used as JMF/@SenderID in Messages that the Controller sends, and SHOULD be used as JMF/@DeviceID for Messages intended for this Controller.
URL		w			r?		
DescriptiveName		w			r?		Any descriptive information that could be used by a person to help determine which Controller a new Device should be associated with.

6.4 KnownDevices

Note: In this section, the Conformance-Table columns for Manager and Worker are relabeled ***Producer*** and ***Consumer***. See section 5.1 Manager/Worker versus Producer/Consumer.

6.4.1 Query – KnownDevices

Table 27: Query – KnownDevices

Instance of: Query

In: List of JMF Messages

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
KnownDevices		w			r		
xsi:type		w			r?		
QueryKnownDevices		w			r?		
DeviceFilter		w			r		See Table 28: DeviceFilter.

6.4.1.1 DeviceFilter

Table 28: DeviceFilter

Referenced by: Query – KnownDevices

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
DeviceDetails		w			r		**r-Test:** The information in the Response contains the information corresponding to the value of *DeviceDetails*.
None		w←			r		Provide only DeviceInfo/@DeviceID and DeviceInfo/@DeviceStatus.
Brief		w←			r		
Details		w←			r		

6.4.2 Response – KnownDevices

Note: the Producer in the following Response tables is returning the response to the Consumer. The Query Producer is the Response Consumer, and the Response Producer is the Query Consumer.

Table 29: Response – KnownDevices

Instance of: Response

In: List of JMF Messages

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
KnownDevices		w			r		
xsi:type		w			r?		
ResponseKnownDevices		w			r?		
DeviceList		w			r		See Table 30: DeviceList.

6.4.2.1 DeviceList

Table 30: DeviceList
Referenced by: Response – KnownDevices

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
DeviceInfo		w			r		See Table 31: DeviceInfo.

6.4.2.2 DeviceInfo

Table 31: DeviceInfo
Referenced by: DeviceList

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
DeviceID		w			r?		
DeviceStatus		w			r?		
Device		w			r?		There MUST be at least one **Device** Element present where Device/@DeviceID matches JMF/@SenderID for the KnownDevices Response. See Table 32: Device.

6.4.2.3 Device

Table 32: Device
Referenced by: DeviceInfo

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
DescriptiveName		w			r?		Any descriptive information that helps identify the Device/Controller.
DeviceID		w			r?		
JDFVersions		w			r?		

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
JMFSenderID		w			r?		If a Device sends its own JMF Messages, the value of *JMFSenderID* MUST match the value of *DeviceID*. Controllers MUST specify their *ControllerID* value in this Attribute.
JMFURL		w←			r?		*JMFURL* MUST be written for all Devices that Support JMF, including the Device whose *DeviceID* matches the *SenderID* in the KnownDevices Response.

6.5 KnownMessages

Note: In this section, the Conformance-Table columns for Manager and Worker are relabeled ***Producer*** and ***Consumer***. See section 5.1 Manager/Worker versus Producer/Consumer.

6.5.1 Query – KnownMessages

The KnownMessages Query is the recommended way to “ping” another Device/Controller, even if you aren’t really interested in the Messages that are Supported. It is a good way to check if configuration is correct.

Table 33: Query – KnownMessages

Instance of: Query

In: List of JMF Messages

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
KnownMessages		w			r		
xsi:type		w			r?		
QueryKnownMessages		w			r?		
KnownMsgQuParams		w			r		See Table 34: KnownMsgQuParams.

6.5.1.1 KnownMsgQuParams

Table 34: KnownMsgQuParams

Referenced by: Query – KnownMessages

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
Exact		w←			r		
false		w←			r		**r-Test:** Response MUST NOT contain DevCaps for the Messages.
true		w?			r?		Implementing Capabilities in KnownMessages is optional at all levels.
ListCommands		w?			r		**r-Test:** If true, Response MUST include Supported commands.
ListQueries		w?			r		**r-Test:** If true, Response MUST include Supported queries.
ListRegistrations		w?			r		**r-Test:** If true, Response MUST include Supported registrations.
ListSignals		w?			r		**r-Test:** If true, Response MUST include Supported signals.
Persistent		w?			r		**r-Test:** If true, Response MUST include only those messages that can use Persistent Channels.

6.5.2 Response – KnownMessages

Note: the Producer in the following Response tables is returning the response to the Consumer. The Query Producer is the Response Consumer, and the Response Producer is the Query Consumer.

Table 35: Response – KnownMessages

Instance of: Response

In: List of JMF Messages

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
KnownMessages		w			r		
xsi:type		w			r?		
ResponseKnownMessages		w			r?		
MessageService		w			r		See Table 36: MessageService.

6.5.2.1 MessageService

Table 36: MessageService
Referenced by: Response – KnownMessages

Name or Value	*Producer*			*Consumer*			Description
Level ➔	1	2	3	1	2	3	
Acknowledge		w			r?		
Command		w			r?		
JMFRole		w			r?		
Persistent		w			r?		
Query		w			r?		
Registration		w			r?		
Signal		w			r?		
Type		w			r?		
URLSchemes		w			r?		

6.6 Occupation

6.6.1 Query – Occupation

Table 37: Query – Occupation
Instance of: Query
In: List of JMF Messages

Name or Value	*Producer*			*Consumer*			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
Occupation		w			r		
xsi:type		w			r?		
QueryOccupation		w			r?		
EmployeeDef		w?			r		See Table 38: EmployeeDef.

6.6.1.1 EmployeeDef

Table 38: EmployeeDef
Referenced by: Query – Occupation

Name or Value	Producer			Consumer			Description
Level →	1	2	3	1	2	3	
PersonalID		w			r		**r-Test:** Response MUST include only information about the employee identified by *PersonalID*.

6.6.2 Response – Occupation

Note: the Producer in the following Response tables is returning the response to the Consumer. The Query Producer is the Response Consumer, and the Response Producer is the Query Consumer.

Table 39: Response – Occupation
Instance of: Response
In: List of JMF Messages

Name or Value	Producer			Consumer			Description
Level →	1	2	3	1	2	3	
Type		w			r		
Occupation		w			r		
xsi:type		w			r?		
ResponseOccupation		w			r?		
Occupation		w			r		See Table 40: Occupation.

6.6.2.1 Occupation

Table 40: Occupation
Referenced by: Response – Occupation

Name or Value	Producer			Consumer			Description
Level →	1	2	3	1	2	3	
Busy		w			r?		
JobID		w←			r?		If the employee referenced in the **Employee** Element is working on a specific Job, *JobID* MUST be specified.

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
JobPartID		w←			r?		If the employee referenced in the **Employee** Element is working on a specific Node of a Job, *JobPartID* MUST be specified.
QueueEntryID		w←			r?		If the employee referenced in the **Employee** Element is working on a specific Queue entry, *QueueEntryID* MUST be specified.
Employee		w			r?		See Table 41: Employee.
`Part`		w←			r?		If the employee is working on one, some, but not all Partitions of a Node of a Job, the `Part` Element MUST be specified with appropriate Partition Keys specified to identify the Partition(s) that the employee is working on. See [JDF1.3].

6.6.2.2 Employee

Table 41: Employee

Referenced by: Occupation

Name or Value	Producer			Consumer			Description
Level ➔	1	2	3	1	2	3	
PersonalID		w			r?		
Roles		w			r?		
Shift		w←			r?		If the shift is known for the employee, it MUST be specified.
CostCenter		w←			r?		If the MIS Cost Center is known for the employee, it MUST be specified. See Table 42: CostCenter.
Person		w?			r?		See Table 43: Person.

6.6.2.3 CostCenter

Table 42: CostCenter
Referenced by: Employee

Name or Value	*Producer*			*Consumer*			Description
Level ➔	1	2	3	1	2	3	
CostCenterID		w			r?		

6.6.2.4 Person

Table 43: Person
Referenced by: Employee

Name or Value	*Producer*			*Consumer*			Description
Level ➔	1	2	3	1	2	3	
FamilyName		w←			r?		If the employee's family name is known, it MUST be specified.
FirstName		w←			r?		If the employee's first name is known, it MUST be specified.
DescriptiveName		w←			r?		If *FamilyName* and *FirstName* are not specified, the employee's full name MUST be specified in the *DescriptiveName* Attribute.

6.7 QueueStatus

6.7.1 Query – QueueStatus

Table 44: Query – QueueStatus
Instance of: Query
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
QueueStatus		w			r		
xsi:type		w			r?		
QueryQueueStatus		w			r?		

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
QueueFilter		w			r		See Table 17: QueueFilter.

6.7.2 Response – QueueStatus

Table 45: Response – QueueStatus

Instance of: Response

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
QueueStatus		r			w		
xsi:type		r?			w		
ResponseQueueStatus		r?			w		
Queue		r			w		See Table 19: Queue.

6.8 RemoveQueueEntry

6.8.1 Command – RemoveQueueEntry

Table 46: Command – RemoveQueueEntry

Instance of: Command

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
RemoveQueueEntry		w			r		
xsi:type		w			r?		
CommandRemoveQueueEntry		w			r?		
QueueEntryDef		w			r		See Table 15: QueueEntryDef.
QueueFilter		w			r		See Table 17: QueueFilter.

6.8.2 Response – RemoveQueueEntry

Table 47: Response – RemoveQueueEntry
Instance of: Response
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
RemoveQueueEntry		r			w		
xsi:type		r?			w		
ResponseRemoveQueueEntry		r?			w		
Queue		r			w		See Table 19: Queue.

6.9 RequestQueueEntry

6.9.1 Command – RequestQueueEntry

Table 48: Command – RequestQueueEntry
Instance of: Command
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
RequestQueueEntry		r			w		
xsi:type		r?			w		
CommandRequestQueueEntry		r?			w		
RequestQueueEntryParams		r			w		See Table 49: RequestQueueEntryParams.

6.9.1.1 RequestQueueEntryParams

Table 49: RequestQueueEntryParams

Referenced by: Command – RequestQueueEntry

Name or Value	Manager			Worker			Description
Level →	1	2	3	1	2	3	
JobID		r			w		**r-Test:** If the Response to the RequestQueueEntry specifies *ReturnCode* = "0", the Manager MUST initiate a SubmitQueueEntry Message for the Job specified by *JobID*.
JobPartID		r			w?		**r-Test:** If the Response to the RequestQueueEntry specifies *ReturnCode* = "0", the Manager MUST initiate a SubmitQueueEntry Message for the Node specified by *JobPartID*.
QueueURL		r			w		**r-Test:** If the Response to the RequestQueueEntry specifies *ReturnCode* = "0", the Manager MUST initiate a SubmitQueueEntry Message to the URL specified by *QueueURL*.
Part		r			w?		**r-Test:** If the Response to the RequestQueueEntry specifies *ReturnCode* = "0", the Manager MUST initiate a SubmitQueueEntry Message for the Partition of the Node identified by the Part Elements. See [JDF1.3].

6.9.2 Response – RequestQueueEntry

Table 50: Response – RequestQueueEntry

Instance of: Response

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level →	1	2	3	1	2	3	
Type		w			r		
RequestQueueEntry		w			r		
xsi:type		w			r?		
ResponseRequestQueueEntry		w			r?		

6.10 ResubmitQueueEntry

6.10.1 Command – ResubmitQueueEntry

Table 51: Command – ResubmitQueueEntry

Instance of: Command

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	**1**	**2**	**3**	**1**	**2**	**3**	
Type		w			r		
ResubmitQueueEntry		w			r		
xsi:type		w			r?		
CommandResubmitQueueEntry		w			r?		
ResubmissionParams		w			r		See Table 52: ResubmissionParams.
QueueFilter		w			r		See Table 17: QueueFilter.

6.10.1.1 ResubmissionParams

Table 52: ResubmissionParams

Referenced by: Command – ResubmitQueueEntry

Name or Value	Manager			Worker			Description
Level ➔	**1**	**2**	**3**	**1**	**2**	**3**	
QueueEntryID		w			r		**r-Test:** The QueueEntry identified by *QueueEntryID* (and no other) is updated.
URL		w			r		**r-Test:** The QueueEntry is updated based on the referenced JDF Instance.

6.10.2 Response – ResubmitQueueEntry

Table 53: Response – ResubmitQueueEntry
Instance of: Response
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
ResubmitQueueEntry		r			w		
xsi:type		r?			w		
ResponseResubmitQueueEntry		r?			w		
Queue		r			w		See Table 19: Queue.

6.11 ResumeQueueEntry

6.11.1 Command – ResumeQueueEntry

Table 54: Command – ResumeQueueEntry
Instance of: Command
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
AcknowledgeURL		w←			r?		If a Manager allows the Worker to send an Acknowledge, the Manager MUST supply this Attribute. A Worker MAY respond synchronously if it can resume Queue entries within the time frame of an http connection. If a Worker chooses to acknowledge a ResumeQueueEntry, it MUST send an Acknowledge Message once the Queue entry has been resumed.
Type		w			r		
ResumeQueueEntry		w			r		
xsi:type		w			r?		
CommandResumeQueueEntry		w			r?		
QueueEntryDef		w			r		See Table 15: QueueEntryDef.

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
QueueFilter		w			r		See Table 17: QueueFilter.

6.11.2 Response – ResumeQueueEntry

Table 55: Response – ResumeQueueEntry

Instance of: Response

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Acknowledged		r?			w←		
true		r?			w←		
false		r?			w←		
Type		r			w		
ResumeQueueEntry		r			w		
xsi:type		r?			w		
ResponseResumeQueueEntry		r?			w		
Queue		r			w←		If Response/@Acknowledged=false, Queue MUST be written. Otherwise Queue MAY be written. See Table 19: Queue.

6.11.3 Acknowledge – ResumeQueueEntry

Table 56: Acknowledge – ResumeQueueEntry

Instance of: Acknowledge

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
ResumeQueueEntry		r			w		
xsi:type		r?			w		
AcknowledgeResumeQueueEntry		r?			w		

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Queue		r			w←		If the Acknowledge Message is the final Acknowledge Message to the ResumeQueueEntry Command, Queue MUST be written. Otherwise Queue MAY be written. See Table 19: Queue.

6.12 ReturnQueueEntry

6.12.1 Command – ReturnQueueEntry

Table 57: Command – ReturnQueueEntry

Instance of: Command

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
AcknowledgeURL		r?			w?		A Manager MAY respond synchronously if it accepts returned Queue entries and parses the JDF within the time frame of an http connection. If a Manager chooses to acknowledge a ReturnQueueEntry, it SHOULD send an Acknowledge Message in a timely fashion.
Type		r			w		
ReturnQueueEntry		r			w		
xsi:type		r?			w		
CommandReturnQueueEntry		r?			w		
ReturnQueueEntryParams		r			w		See Table 58: ReturnQueueEntryParams.

6.12.1.1 ReturnQueueEntryParams

Table 58: ReturnQueueEntryParams
Referenced by: Command – ReturnQueueEntry

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Aborted		r?			w←		
Completed		r?			w←		
QueueEntryID		r?			w		*QueueEntryID* was added as an errata to [JDF1.3].
URL		r?			w		References JDF Instance.
cid:…		r			w		URL whose scheme is "*cid*".

6.12.2 Response – ReturnQueueEntry

Table 59: Response – ReturnQueueEntry
Instance of: Response
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Acknowledged		w←			r?		
true		w←			r?		
false		w←			r?		
Type		w			r		
ReturnQueueEntry		w			r		
xsi:type		w			r?		
ResponseReturnQueueEntry		w			r?		

6.12.3 Acknowledge – ReturnQueueEntry

Table 60: Acknowledge – ReturnQueueEntry

Instance of: Acknowledge

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
ReturnQueueEntry		w			r		
xsi:type		w			r?		
AcknowledgeReturnQueueEntry		w			r?		

6.13 SetQueueEntryPosition

6.13.1 Command –SetQueueEntryPosition

Table 61: Command – SetQueueEntryPosition

Instance of: Command

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
SetQueueEntryPosition		w			r		
xsi:type		w			r?		
CommandSetQueueEntryPosition		w			r?		
QueueEntryPosParams		w			r		See Table 62: QueueEntryPosParams.
QueueFilter		w			r		See Table 17: QueueFilter.

6.13.1.1 QueueEntryPosParams

Table 62: QueueEntryPosParams

Referenced by: Command – SetQueueEntryPosition

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
NextQueueEntryID		w←			r		Exactly one of *NextQueueEntryID*, *PrevQueueEntryID*, or *Position* MUST be specified. **r-Test:** The QueueEntry identified by *QueueEntryID* is positioned immediately prior to the QueueEntry identified by *NextQueueEntryID*.
QueueEntryID		w			r		**r-Test:** The Queue returned in the response has moved the QueueEntry identified by *QueueEntryID* according to the other Attributes.
PrevQueueEntryID		w←			r		Exactly one of *NextQueueEntryID*, *PrevQueueEntryID*, or *Position* MUST be specified. **r-Test:** The QueueEntry identified by *QueueEntryID* is positioned immediately after to the QueueEntry identified by *PrevQueueEntryID*.
Position		w←			r		Exactly one of *NextQueueEntryID*, *PrevQueueEntryID*, or *Position* MUST be specified. **r-Test:** The QueueEntry identified by *QueueEntryID* is in the Queue in the position specified by *Position*, where the first QueueEntry is in position 0.

6.13.2 Response – SetQueueEntryPosition

Table 63: Response – SetQueueEntryPosition

Instance of: Response

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
SetQueueEntryPosition		r			w		
xsi:type		r?			w		
ResponseSetQueueEntryPosition		r?			w		
Queue		r			w		See Table 19: Queue.

6.14 SetQueueEntryPriority

6.14.1 Command –SetQueueEntryPriority

Table 64: Command – SetQueueEntryPriority
Instance of: Command
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
SetQueueEntryPriority		w			r		
xsi:type		w			r?		
CommandSetQueueEntryPriority		w			r?		
QueueEntryPriParams		w			r		See Table 65: QueueEntryPriParams.
QueueFilter		w			r		See Table 17: QueueFilter.

6.14.1.1 QueueEntryPriParams

Table 65: QueueEntryPriParams
Referenced by: Command – SetQueueEntryPriority

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Priority		w			r		**r-Test:** In the Queue returned in the response, the QueueEntry identified by *QueueEntryID* has the priority indicated.
QueueEntryID		w			r		**r-Test:** In the Queue returned in the response,

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							the QueueEntry identified by *QueueEntryID* has the priority indicated.

6.14.2 Response – SetQueueEntryPriority

Table 66: Response – SetQueueEntryPriority

Instance of: Response

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
SetQueueEntryPriority		r			w		
xsi:type		r?			w		
ResponseSetQueueEntryPriority		r?			w		
Queue		r			w		See Table 19: Queue.

6.15 StopPersistentChannel

6.15.1 Command – StopPersistentChannel

Table 67: Command – StopPersistentChannel

Instance of: Command

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
StopPersistentChannel		w			r		
xsi:type		w			r?		
CommandStopPersistentChannel		w			r?		
StopPersChParams		w			r		See Table 68: StopPersChParams.

6.15.1.1 StopPersChParams

Table 68: StopPersChParams

Referenced by: Command – StopPersistentChannel

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
ChannelID		w?			r		The /JMF/Query/@ID in the Query Message that the Manager sent to create a Persistent Channel. If *ChannelID* is specified, *DeviceID* SHOULD also be specified. **r-Test:** No further signals or commands are created by the Worker that specifies a value of /JMF/Query/@refID that matches *ChannelID*. See 5.4.1.3 for more information.
DeviceID		w?			r		**r-Test:** No further signals are received that specify a value of /JMF/ @SenderID that matches *DeviceID*. See 5.4.1.3 for more information.
JobID		w←			r		If *JobPartID* is specified, *JobID* MUST be specified. If only *JobID* is specified, closes Persistent Channels that were setup for the specified *JobID*. **r-Test:** No further signals are created by the Worker that reference the specified *JobID*. See 5.4.1.3 for more information.
JobPartID		w?			r		If specified, closes Persistent Channels that were setup for the specified *JobID* and/or *JobPartID* combination. If *JobPartID* is specified, *JobID* SHOULD also be specified. **r-Test:** No further signals are created by the Worker that references the specified *JobID* and/or *JobPartID* combination. See 5.4.1.3 for more information.
QueueEntryID		w?			r		If specified, closes Persistent Channels that were setup for the specified *QueueEntryID*. **r-Test:** No further signals are created by the Worker that reference the specified *QueueEntryID*. See 5.4.1.3 for more information.

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
URL		w			r		URL of the receiver of the Messages. **r-Test:** No further signals are created by the Worker that were setup to be sent to the specified URL. See section 5.4.1.3 for more information.
http:…		w			r		URL whose scheme is "*http*".
all remaining values	!w			r?			

6.15.2 Response – StopPersistentChannel

Table 69: Response – StopPersistentChannel
Instance of: Response
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
StopPersistentChannel		r			w		
xsi:type		r?			w		
ResponseStopPersistentChannel		r?			w		

6.16 SubmissionMethods

6.16.1 Query – SubmissionMethods

Table 70: Query – SubmissionMethods
Instance of: Query
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		w			r		
SubmissionMethods		w			r		
xsi:type		w			r?		

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
QuerySubmissionMethods		w			r?		

6.16.2 Response – SubmissionMethods

Table 71: Response – SubmissionMethods
Instance of: Response
In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Type		r			w		
SubmissionMethods		r			w		
xsi:type		r?			w		
ResponseSubmissionMethods		r?			w		
SubmissionMethods		r			w		See Table 72: SubmissionMethods.

6.16.2.1 SubmissionMethods

Table 72: SubmissionMethods
Referenced by: Response – SubmissionMethods

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
HotFolder		r?			!w		If submission is Supported via Hot Folder, the URL can be found via the KnownDevices Response (in DeviceList/DeviceInfo/Device/@JMFURL) or has to be configured manually.
Packaging		r?			w		If no types of packaging are Supported, an empty *Packaging* Attribute MUST be written.
URLSchemes		r?			w		If retrieving referenced files is not Supported, an empty *URLSchemes* Attribute MUST be written.

6.17 SubmitQueueEntry

6.17.1 Command – SubmitQueueEntry

Table 73: Command – SubmitQueueEntry

Instance of: Command

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
AcknowledgeURL		w			r?		A Worker MAY respond synchronously if it accepts Queue entries and parses the JDF within the time frame of an http connection. If a Worker chooses to acknowledge a SubmitQueueEntry, it SHOULD send an Acknowledge Message in a timely fashion and not wait until it executes the Node.
Type		w			r		
SubmitQueueEntry		w			r		
xsi:type		w			r?		
CommandSubmitQueueEntry		w			r?		
QueueSubmissionParams		w			r		See Table 74: QueueSubmissionParams.
QueueFilter		w←			r		It is highly recommended that the Manager supply this Element. See Table 17: QueueFilter.

6.17.1.1 QueueSubmissionParams

Table 74: QueueSubmissionParams

Referenced by: Command – SubmitQueueEntry

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Hold		w?			r←		If the Worker Supports the HoldQueueEntry Message, the Worker MUST Support this *Hold* Attribute. **r-Test:** If QueueSubmissionParams/ @Hold is true, then immediately after submission QueueEntry/@Status MUST be

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							"`Held`".
Priority		w?			r←		If the Worker Supports the `SetQueueEntryPriority` Message, the Worker MUST Support this *Priority* Attribute. **r-Test:** If this Attribute is provided, then immediately after submission `QueueEntry/@Priority` MUST equal `QueueSubmissionParams/@Priority`.
ReturnJMF		w			r		**r-Test:** When the `QueueEntry` is complete, the Worker sends the `ReturnQueueEntry` to the URL specified by `@ReturnJMF`.
`http:…`		w			r		URL whose scheme is "`http`".
ReturnURL		!w			r?		
URL		w			r		References a JDF Instance. The Manager SHOULD use a scheme of "`cid`". If the scheme is not "`cid`" for the value of this Attribute, the Manager MUST keep the JDF available for the Worker to retrieve until the Worker completes or aborts the Job as indicated by sending the updated JDF with a `ReturnQueueEntry` Command. **r-Test:** The Worker successfully retrieves the JDF instance via the URL.
`cid:…`		w←			r		URL whose scheme is "`cid`". This is the recommended URL scheme for Managers to use.
`file:…`		w←			r		URL whose scheme is "`file`".
`ftp:…`		w?			r?		URL whose scheme is "`ftp`".
`http:...`		w←			r		URL whose scheme is "`http`".
all remaining values	!w			r?			
Comment		w?			r		See Table 16: Comment.

6.17.2 Response – SubmitQueueEntry

The Worker MUST return a `SubmitQueueEntry Response` before the HTTP connection would time out. In addition, the Worker MUST parse the JDF supplied in the `SubmitQueueEntry Command`. If the Worker is unable to parse the JDF before returning the `SubmitQueueEntry Response`, the Worker MUST return the `Response` followed by a `SubmitQueueEntry Acknowledge` after the Worker has parsed the JDF (see section 5.4.2 “Asynchronous Acknowledges” and Table 76: Acknowledge – SubmitQueueEntry).

Table 75: Response – SubmitQueueEntry

Instance of: Response

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level →	1	2	3	1	2	3	
Acknowledged		r			w←		
true		r			w←		
false		r			w←		
Type		r			w		
SubmitQueueEntry		r			w		
xsi:type		r?			w		
ResponseSubmitQueueEntry		r?			w		
QueueEntry		r			w←		The Worker MUST write this Element unless the Command is being acknowledged (@Acknowledged = "true"), or the Command failed (@ReturnCode != "0"). See Table 20: QueueEntry.
Queue		r?			w←		The Worker MUST write this Element unless the Command is being acknowledged (@Acknowledged = "true"). See Table 19: Queue.

6.17.3 Acknowledge – SubmitQueueEntry

Table 76: Acknowledge – SubmitQueueEntry

Instance of: Acknowledge

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level →	1	2	3	1	2	3	
Type		r			w		
SubmitQueueEntry		r			w		
xsi:type		r?			w		
AcknowledgeSubmitQueueEntry		r?			w		
QueueEntry		r			w←		The Worker MUST NOT supply this

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							Element if the Command fails. See Table 20: QueueEntry.
Queue		r?			w		See Table 19: Queue.

6.18 SuspendQueueEntry

6.18.1 Command – SuspendQueueEntry

Table 77: Command – SuspendQueueEntry

Instance of: Command

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
AcknowledgeURL		w←			r?		If a Manager allows the Worker to send an Acknowledge, the Manager MUST supply this Attribute. A Worker MAY respond synchronously if it can suspend Queue entries within the time frame of an http connection. If a Worker chooses to acknowledge a SuspendQueueEntry, it MUST send an Acknowledge Message once the Queue entry has been suspended.
Type		w			r		
SuspendQueueEntry		w			r		
xsi:type		w			r?		
CommandSuspendQueueEntry		w			r?		
QueueEntryDef		w			r		See Table 15: QueueEntryDef.
QueueFilter		w			r		See Table 17: QueueFilter.

6.18.2 Response – SuspendQueueEntry

Table 78: Response – SuspendQueueEntry

Instance of: Response

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level →	1	2	3	1	2	3	
Acknowledged		r?			w←		
true		r?			w←		
false		r?			w←		
Type		r			w		
SuspendQueueEntry		r			w		
xsi:type		r?			w		
ResponseSuspendQueueEntry		r?			w		
Queue		r			w←		If Response/@Acknowledged = "false", Queue MUST be written. Otherwise Queue MAY be written. See Table 19: Queue.

6.18.3 Acknowledge – SuspendQueueEntry

Table 79: Acknowledge – SuspendQueueEntry

Instance of: Acknowledge

In: List of JMF Messages

Name or Value	Manager			Worker			Description
Level →	1	2	3	1	2	3	
Type		r			w		
SuspendQueueEntry		r			w		
xsi:type		r?			w		
AcknowledgeSuspendQueueEntry		r?			w		
Queue		r			w←		If the Acknowledge Message is the final Acknowledge Message to the SuspendQueueEntry Command, Queue MUST be written. Otherwise Queue MAY be written. See Table 19: Queue.

7 Conformance Rules – Job Submission

7.1 Plain JDF versus JMF – SubmitQueueEntry

7.1.1 SubmitQueueEntry, QueueEntry, ReturnQueueEntry Relationship

A JDF Job submission via the SubmitQueueEntry Command MUST result in a single QueueEntry being created by the Worker, and the Worker MUST send at least one ReturnQueueEntry Message when the QueueEntry is completed or aborted. The Worker MAY send multiple ReturnQueueEntry Messages.

When a Worker determines that further work needs to be done on a Completed QueueEntry, the Worker MUST set the QueueEntry/@Status to "*Waiting*" or "*Running*", and MUST send any signals associated with this change of status, and MUST reactivate any Persistent Channels related to JMF Queries embedded in the JDF.

The Manager MUST Support receiving multiple ReturnQueueEntry Messages related to the same QueueEntry from the Worker. If the Manager needs to prevent further ReturnQueueEntry Messages related to the same QueueEntry from being sent by the Worker, the Manager MUST send a RemoveQueueEntry to remove the related QueueEntry.

If a QueueEntry is removed while in the *Waiting* or *Held* state using the RemoveQueueEntry Command, a ReturnQueueEntry Message NEED NOT be sent.

If a QueueEntry is removed while in the *Completed* or *Aborted* state using the RemoveQueueEntry Command, further ReturnQueueEntry Messages MUST NOT be sent. This ICS enforces a restriction of a single SubmitQueueEntry Command per JMF Message. A JMF Element containing a SubmitQueueEntry Command MUST contain one and only one SubmitQueueEntry Command Message.

7.1.2 URL External Reference versus MIME Encoded

When a Manager submits a JDF Instance via a JMF SubmitQueueEntry Command Message, the Manager MUST use one of these options:

A. The JDF Instance and the JMF Message are packaged together using MIME. The JMF Message MUST be the first part of the MIME package, and MUST use a URL with the "*cid*" scheme to reference the JDF part of the MIME package. This is the recommended option.
B. The JDF Instance is separate from the JMF Message. The JMF Message uses a URL to reference the JDF Instance.

To be conformant to Level 2 of this ICS, Managers and Workers MUST Support both option A and option B.

8 References

8.1 Normative References

[Base-ICS] Base ICS, Version 1.3, published July 2007. Available at: http://www.cip4.org.

[JDF1.3] JDF Specification, Version 1.3, published September 30, 2005, and Errata, JDF Specification, Version 1.3. Available at: http://www.cip4.org.