Base ICS

Version 1.5

Date: 2015-05-11

File: ICS-Base-1.5-150511.docx, *.pdf*

System Behavior and Interoperability WG

Abstract

This document is the first of a series of ***ICS*** (Interoperability Conformance Specification) documents. Each ***ICS*** defines a set of ***Conformance Requirements*** that a conforming JDF-enabled Product SHALL meet in order to achieve interoperability with other conforming JDF-enabled Products. This document, the Base ICS, defines the ***Conformance Requirements*** that are common to all other ***ICSs***, i.e. those ***Conformance Requirements*** that have been factored out of the other ***ICSs*** and placed in this ***ICS***. This document specifies two ***Conformance Levels*** of ***Conformance Requirements***. These levels mainly differ in the method of providing / accessing ***Referenced Files*** between the ***Manager*** and the ***Worker*** and include the ***Conformance Requirements*** for ***Hot Folders***, string Attributes, and specific JDF ***Elements***.

This version applies to interactions using [JDF1.5].

Copyright Notice

Table of Contents

Figures

Tables

1 Introduction

Because the JDF specification [JDF1.5][1] is quite large, it is unrealistic (and not very useful) for any JDF-enabled Product to implement the [JDF1.5] in full. Yet, if each JDF-enabled Product were to implement an arbitrary subset of the [JDF1.5], interoperability between JDF-enabled Products would be highly unlikely.

Hence, there is a need for a number of well-specified subsets of JDF, each defining an Interface between pairs of vendor's Products in the workflow. The mechanism for specifying such a subset of JDF is the ***Interoperability Conformance Specification (ICS)***. An ***ICS*** defines a subset of JDF by means of ***Conformance Requirements***, which are a set of requirements. When a JDF-enabled Product meets the ***Manager Conformance Requirement*** of a particular ***ICS***, it achieves interoperability with other JDF-enabled Products that meet the corresponding ***Worker Conformance Requirements*** of the same ***ICS***.

Note: The definitions of capitalized terms appear in section 2 Glossary below.

2 Glossary

This section defines terminology used throughout ***ICSs***. The terms appear in alphabetic order. If a word is in ***bold-italic***, it is defined elsewhere in this section. Elsewhere in ***ICSs***, the first letter of each word of these terms is capitalized.

Table 1: Glossary

Term	Definition
Abstract Element	An ***Element*** that is a placeholder for other ***Elements*** and may describe ***Traits*** that are common to other ***Elements***. Such other ***Elements*** are said to be derived from the ***Abstract Element***. For example, Audit is an ***Abstract Element***. The Created ***Element*** and Modified ***Element*** are both derived from the Audit ***Abstract Element***. An ***Abstract Element*** does not appear in a ***JDF Instance***.
Agent	See [JDF1.5]Section 1.4 "Glossary" and see ***Producer*** in this Table 1: Glossary.
Attribute	See [JDF1.5] Section 1.4 "Glossary".
Attribute Value	The value of an ***Attribute***.
Conformance Level	Defines a subset of ***Conformance Requirements*** for an ***ICS***. Level-1***Conformance Requirements*** from a subset of Level-2 ***Conformance Requirements***, and so on for higher levels.
Conformance Requirement	A single requirement that a conforming JDF-enabled Product SHALL meet. An ***ICS*** specifies a set of ***Conformance Requirements*** that a conforming JDF-enabled Product SHALL meet in order to achieve interoperability with other conforming JDF-enabled Products that meet the same ***Conformance Requirements***.
Conformance Table	Describes the ***Conformance Requirements*** for a single ***Element*** of a ***JDF Instance*** or ***JMF Message***. Each row of a ***Conformance Table*** contains a single ***Trait*** of the ***Element***. Each such ***Trait*** is subject to two ***Conformance Requirements***, one that applies to a conforming ***Manager*** and another that applies to a conforming ***Worker***.
Consumer	A ***Manager*** or ***Worker*** in a role where it consumes a ***JDF Instance*** or ***JMF***

[1]The bracketed [JDF1.5] serves as a reference to Section 10.1 *Normative References*. It can also be the document's name, as in "The [JDF1.5] …", which means "The JDF Specification, version 1.5 …".

Term	Definition
	Message, i.e. reads and processes a JDF Instance or ***JMF Message***. See ***Producer*** in this section. See also "**JDF Consumer**" in [JDF1.5] Section 1.4 "Glossary".
Controller	See [JDF1.5] Section 1.4 "Glossary".
Derived Element	An Element that is based on some ***Abstract Element***. See ***Abstract Element*** for an example.
Device	See [JDF1.5] Section 1.4 "Glossary".
Device Worker	The ***Worker*** part of a ***Device***.
Element	See [JDF1.5] Section 1.4 "Glossary".
Hot Folder	A folder that is watched by the ***Worker***, so that when the ***Manager*** writes a file into the ***Hot Folder***, the ***Worker*** interprets that action as a Job submission and attempts to perform the actions specified in the ***JDF Instance*** or ***JMF Message*** contained in the file.
Interoperability Conformance Specification (ICS)	A specification developed by a CIP4 WG and approved by the CIP4 Technical Steering Committee (TSC). An ICS specifies the ***Manager Conformance Requirements*** for an interface that a conforming JDF-enabled Product SHALL meet in order to achieve interoperability with other conforming JDF-enabled Products that meet the corresponding ***Worker Conformance Requirements***.
JDF	See [JDF1.5] Section 1.4 "Glossary".
JDF Instance	An XML document that is a valid JDF ***Node*** according to [JDF1.5]. The JDF ***Node*** describes a print Job or some portion thereof.
JDF Instance File	A file that contains a ***JDF Instance*** only.
JDF MIME Instance	A MIME Multipart/Related data stream [RFC 2387] whose first body part is a ***JDF Instance*** and each remaining body part is identified by a *Content-ID Header* [RFC 2392]and is referenced from the ***JDF Instance*** body part using a "`cid`" URL [RFC 2392].
JDF MIME File	A file that contains a ***JDF MIME Instance***.
JMF	See [JDF1.5] Section 1.4 "Glossary".
JMF Message	An XML document that is a valid JMF ***Element*** according to the JDF Schema.
Manager	The software that implements the ***Manager Interface***.
Manager Interface	The interface that sends ***JDF Instances***, ***JMF Messages*** and other data (possibly via the network) to a ***Worker*** in a ***Device*** or ***Controller*** in the hierarchy below (see [JDF1.5] Figure 2.1 "Example of JDF and JMF workflow interactions") and may receive information back (possibly via the network) from a ***Worker*** in a ***Device*** or ***Controller***.
MAY	See [JDF1.5] Section 1.4.1 Conformance Terminology".
MIS	See [JDF1.5] Section 1.4 "Glossary".
MIS Manager	The ***Manager Interface*** of the ***MIS***.
NEED NOT	Indicates an action that is not required for conformance, but MAY be performed. See [JDF1.5] Section 1.4.1 Conformance Terminology".
Node	See [JDF1.5] Section 1.4 "Glossary".

Term	Definition
Process	See [JDF1.5] Section 1.4 “Glossary”.
Producer	A ***Manager*** or ***Worker*** in a role where it produces or modifies either a ***JDF Instance*** or ***JMF Message***, i.e. writes or updates a ***JDF Instance*** or ***JMF Message***. See ***Agent*** and ***Consumer*** in this section.
Product-Sector ICS	An ***ICS*** that specifies the ***Conformance Requirements*** for a JDF-enabled Product in a specific Product sector. For example, an ***ICS*** for a Product sector that includes binding is likely to have a requirement for a ***Stitching Process***.
Referenced File	A file that is referenced via a URI from a ***JDF Instance*** or a ***JMF Message***. For example, a PDF file could be a ***Referenced File***.
Root Node	The root ***Node***, i.e. the ***Node*** at the top level.
SHALL, SHALL NOT	See [JDF1.5] Section 1.4.1 “Conformance Terminology”.
SHOULD	See [JDF1.5] Section 1.4.1 “Conformance Terminology”.
Subelement	An ***Element*** that is a child of another ***Element***.
Subnode	A ***Node*** that is below the ***Root Node***, i.e. a ***Node*** that is not a ***Root Node***.
Support	See [JDF1.5] Section 1.4 “Glossary”.
Trait	In the context of an ***Element***, a single ***Subelement*** of it, a single ***Attribute*** of it or a single ***Attribute Value*** of one of its ***Attributes***. In the context of the [JDF1.5], a table for an ***Element*** contains all ***Traits*** of the ***Element***.
Worker Interface	The interface that receives ***JDF Instances***, ***JMF Messages*** and other data (possibly via the network) from a ***Manager*** in a ***Controller*** or ***MIS*** in the hierarchy above (see [JDF1.5] Figure 2-1 “Example of JDF and JMF workflow interactions”) and may send information back (possibly via the network) to a ***Manager*** in a ***Controller*** or ***MIS***.
Worker	The software that implements the ***Worker Interface***.

3 Conformance Requirements for all ICSs

This section defines the ***Conformance Requirements*** that apply to any [JDF1.5] based ***ICS***.

A Product conforming to any ***ICS***:

1. SHALL produce ***JDF Instances*** that conform to the [JDF1.5].
2. SHALL meet all appropriate ***Conformance Requirements*** of the respective ***ICS***.
3. SHALL be able to read and write XML encoded in UTF-8 (variable number of octet Unicode characters).

Any claim of conformance to any ***ICS*** by a Product’s Vendor SHALL follow the requirements in the “CIP4 Guidelines: *For CIP4 and JDF Logo Use*” [JDF-logo] for ***ICS*** conformance.

Each ICS document defines ***Conformance Requirements*** for a subset of [JDF1.5].

The remainder of this ICS specifies ***Conformance Requirements*** that are specific to this Base ICS:

4 Conformance Levels

This ICS defines three Conformance Levels, namely Levels 0, 1 and 2. The ***Conformance Requirements*** in this ICS are common to all ***Product-Sector ICSs***.

See Section 2 *Glossary* for definitions of ***Manager***, ***Worker***, ***Producer***, ***Consumer***, *and* ***JDF Instance***.

See Appendix A: "How to Read ICS Documents" 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*** SHALL conform to the ***Manager*** part and a ***Worker*** SHALL 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
0	-	-	-	For unidirectional exchange of JDF from ***Manager*** to ***Worker***. Level 0 is equivalent to Level 1 except for the following points that relate to returning JDF from ***Worker*** to ***Manager***: • A Level 0 ***Manager*** NEED NOT consume ***JDF Instances***. • A Level 0 ***Worker*** NEED NOT produce ***JDF Instances***. • The Write conformance for a ***Worker*** defaults to "`w?`" in section 6 Conformance Tables – JDF Instances. • The Read conformance for a ***Manager*** defaults to "`r?`" in section 6 Conformance Tables – JDF Instances.
1	-	-	-	• The ***Manager*** can generate and receive conformant ***JDF Instance Files***. • The ***Worker*** can accept and (after completion of the Job) return (or forward) conformant ***JDF Instance Files***. • The ***Manager*** and ***Worker*** send and receive ***JDF Instance Files*** via a ***Hot Folder***.
2	1	-	-	• The ***Manager*** can provide access to ***Referenced Files*** via HTTP. • The ***Manager*** can create MIME packages. • The ***Worker*** can retrieve ***Referenced Files*** via HTTP. • The ***Worker*** can interpret MIME packages and access their parts. • The ***Manager*** and ***Worker*** NEED NOT send and receive ***JDF Instance Files*** via a ***Hot Folder***. Note: With respect to transport protocol, this option means that Level 2 requirements are *not* a complete superset of Level 1.

4.1 Availability of Certification

Each ICS includes a table similar to Table 2: Conformance Levels, indicating what levels are required to be Supported in other ICSs. When the the level indicated is just a number, it means that this level SHALL be Supported, and if certification is available for the containing ICS, then certification will ***only*** be available for this specific level of the corresponding ICS.

When the level indicated is a number followed by the words “or higher”, the level number indicated is the minimum level required to be Supported, but that higher levels MAY be Supported and certification is intended to be available for these higher levels. Product vendors will be able to choose whether to certify to higher level, and may choose any level greater than or equal to the required level.

4.2 Interoperability of Levels

Due to the incremental nature of the ***ICS*** levels except for Level 0, a ***Manager*** or ***Worker*** that conforms to a higher level is able to interoperate with a ***Worker*** or ***Manager***, respectively that conforms to a lower level by using this lower level for all communication. This rule does not apply to the transport protocol. Level 2 ***Manager*** and ***Worker*** NEED NOT send and receive ***JDF Instance Files*** via a ***Hot Folder***. Any ICS Feature that is required for ICS conformance MAY be disabled in an implementation-dependent manner. For example, a Base ICS Level 2 Worker implementation MAY disable ***Hot Folders*** (a Base ICS 1.5 Level 1 REQUIRED Feature) and MAY use only HTTPS (disabling HTTP - a Base ICS 1.5 Level 2 Feature), as long as the Worker implementation can be configured to use HTTP (Base ICS 1.5 level 2).

5 Conformance Rules – Values and Extensions

5.1 Data Type Values

This section specifies ***Conformance Requirements*** for certain data type values.

5.1.1 Size Limits for Attribute Values

The length of every ***Attribute Value*** (including list types) except where explicitly excluded, SHALL not exceed 20k characters when the ***Attribute Value*** is encoded as an XML string. This limit ensures that the length of its UTF-8 representation does not exceed 64k.

5.1.1.1 Size Limits for Simple Values

Table 3 specifies the minimum and maximum number of characters allowed for values of each specified data type when such a value occurs in an XML document, i.e. a ***JDF Instance*** or ***JMF Message***.

Note: that this ***ICS*** expresses the maximum size of many strings in terms of *characters*. The number of octets needed per characters varies according to the representation. For example an ASCII character is represented by one octet; a UCS-2 character can be represented directly by two octets or encoded in UTF-8 with 1, 2 or 3 octets depending on the character. Other representations have additional complexity. Thus, if an implementation allocates space for a specified number of octets rather than a specified number of characters, it SHALL know how many octets are needed for each character. Therefore, text data types, except where explicitly excluded, have been limited to 20K (20,480) characters, so that their UTF-8 maximum will never exceed 64K.

Table 3: Minimum and Maximum Number of Characters of Specified JDF Data Types

Data Type	Minimum	Maximum	Description
enumeration	1	63	Although the [JDF1.5] enumerates all values and thus limits what can be written in the JDF, this value is useful for a ***Consumer***.

Data Type	Minimum	Maximum	Description
hexBinary	2	20480	A string of an even number characters restricted to contain digits "0" to "9" and letters "a" to "f" (in lowercase or uppercase) Each pair of characters represents the value of an octet.
ID	1	63	
IDREF	1	63	
NMTOKEN	1	63	
PDFPath	0	***unlimited***	
regExp	1	20480	Regular expression as defined by [XMLSchema].
string	0	1023	A normalizedString [XMLSchema], i.e., character strings without tabs, and line feeds, etc. A number of Attributes that use the string data type MAY be used to index into an internal database. Therefore, such Attributes have a shorter maximum limit than 1023. See Table 5: Attributes with Short String below.
text	0	20480	Text data contained in a telem (text Element).
URI	1	4095	Represents a Uniform Resource Identifier (URI) Reference. See [RFC3986] and [ISO8601:2004].
URL	1	4095	Represents a Uniform Resource Locator (URL) Reference. See [RFC3986] and [ISO8601:2004].
XPath	1	20480	Represents a path to an Element or Attribute in an XML document [XPath].

5.1.1.2 Size Limits for Lists

Table 4 specifies the minimum and maximum number of units allowed for values of each specified data type when such a value occurs in an XML document, i.e. a ***JDF Instance***. Table 4 also specifies the units. For example, NMTOKENS has a minimum length of 0 NMTOKEN's (i.e. 0 octets) and a maximum number of 2048 NMTOKEN's. Because the maximum length of NMTOKEN is 63 and a space character separates each NMTOKEN, the maximum length of NMTOKENS is 2048*(63+1) - 1 = 131,071 characters.

Table 4: Minimum and Maximum Size of List-like JDF Data Types

Data Type	Minimum	Maximum	Units	Description
enumerations	0	63	enumeration	Specifies the number of enumeration's in enumerations. Although the [JDF1.5] enumerates all values and thus limits what can be written in the JDF, this value is useful for a reader application.
NMTOKENS	0	2048	NMTOKEN	Specifies the number of NMTOKEN's in NMTOKENS

Data Type	Minimum	Maximum	Units	Description
SomeTypeList	0	2048	*someType*	For example, if *someType* is IntegerRange, then this rule states that an IntegerRangeList contains at most 2^{11} IntegerRange's, each of which consists of two integers.

5.1.1.3 Size Limits for Attributes with Short Strings

Some string-valued ***Attributes*** identify objects, such as Resources. Such ***Attributes*** have a shorter maximum length of 63 or 255 characters rather than the normal maximum length of 1023 for other string-valued ***Attributes*** because an implementation may use the ***Attribute***'s string value to index into an internal database. Most of the ***Attributes*** in Table 5 below end with "ID".

Table 5: Attributes with Short String

Attribute	Length	Notes
BatchID	63	
CatalogID	63	
CostCenterID	63	
CustomerJobName	255	
CustomerOrderID	63	
CustomerID	63	
DefaultID	63	
DescriptiveName	255	
DeviceID	63	
JMFSenderID	63	
JobID	63	
JobPartID	63	The original @*JobPartID* SHALL NOT exceed 50 characters so that a ***Controller*** can generate nested @*JobPartID* Attributes (when not using NewJDF). For details see @*JobPartID* in Table 7: JDF Node.
LocID	63	
MountID	63	
NextQueueEntryID	63	
PersonalID	63	
PipeID	63	
PrevQueueEntryID	63	
ProductID	63	
ProjectID	63	

Attribute	Length	Notes
QueueEntryID	63	
RelatedJobID	63	
RelatedJobPartID	63	
ResourceID	63	
SenderID	63	
SpawnID	63	
StatusDetails	63	
TemplateID	63	
GeneralID/@*IDUsage*	63	
GeneralID/@*IDValue*	63	

5.1.1.4 Size Limits for Attributes Named in the PartIDKeys Attribute

Table 6 specifies the minimum and maximum size allowed for the values of each specified data type when such a value is the value of a *@PartIDKeys* Attribute in either a Part *Element* or in a Partitioned Resource. See [JDF1.5] Table 3-28: "*Part Element*" for a list of *@PartIDKeys* Attributes and [JDF1.5] Table 3-27 "*Partitionable Resource Element*" for the values of the *@PartIDKeys* Attribute. If a *@PartIDKeys* *Attribute Value* has a data type that is not listed in Table 6, then the data type doesn't have special size limits, and Table 3 and Table 4 specify the size limits.

Table 6: PartIDKeys Attributes – Different Size Limits

Data Type	Minimum	Maximum	Units	Description
string	1	63	character	The number of characters in a string is different.

5.1.2 Format of Values for dateTime Data Type

Values of type dateTime SHALL be in one of the following formats (from [ISO8601:2004]), where the Elements in bold actually appear in the string, and Elements in italics are replaced as indicated:

*YYYY***-***MM***-***DD***T***HH***:***MM***:***SS***[.***ddd***](+|-)***ZZ***:***zz***

or

*YYYY***-***MM***-***DD***T***HH***:***MM***:***SS***[.***ddd***]Z**

where YYYY is the year, MM is the month, DD is the day, followed by the letter 'T', then HH is the hour, MM is the minutes, SS is the seconds, optionally followed by a period/dot and then up to three decimal places for the fractional seconds. Finally, the offset from universal time (UTC) is specified using either a plus or a minus, then ZZ represents the hours of offset, and zz represents the minutes of offset. Note that all hyphens and colons SHALL be specified. The second format, with the Z (which actually appears in the string) excludes the offset from universal time, and indicates that the time is in universal time. Examples of valid dateTime values are:

`2008-12-18T11:55:31-05:00`

`2009-03-25T11:28:49.385+04:30`

2009-06-28T23:18:49Z

5.2 JDF Extensions and JDF 1.5 Deprecated Items

Elements, *Attributes* or *Attribute Values* (including NMTOKEN values) that are undefined in the [JDF1.5] or that are Deprecated in the [JDF1.5] are optional to read or write (w?/r?). That is, write *Support* for JDF extensions and Deprecated *Traits* is allowed. Note that interoperability based on such *Traits* might be reduced.

6 Conformance Tables – JDF Instances

Each table in this section shows the *Conformance Requirements* for an *Element* of a *JDF Instance*.

See Appendix A: for an explanation of the table format and the codes used to specify conformance.

6.1 JDF Node

Table 7 specifies the *Conformance Requirements* for *Attributes* and *Elements* for a JDF *Node*, whether it is a *Root Node* or a *Subnode*. Most of the *Attributes* and *Elements* have the *same Conformance Requirements* whether the *Node* is a *Root Node* or a *Subnode*. Those that differ are marked with "w←" and the Description column specifies the conditions. The JDF *Node* requirements for *Attributes* and *Attribute Values* in Table 6 are fulfilled when the Attributes are specified in the AncestorPool/Ancestor *Elements* and their *Subnodes*, respectively (see Table 15: Ancestor).

Table 7: JDF Node

From: [JDF1.5] Table 3-4

Referenced by: JDF Node

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Activation	w?			r			See [JDF1.5] Section 4.2.1 "Determining Executable Nodes".
Active	w←			r			
CommentURL	w?			r?			Links to a human readable description of the Job in any format. This *ICS* makes no statement about Supported formats or how a *Worker* should deal with the data referenced by the URL. The Job description referenced by this URL does NOT affect the value of any Attributes in the JDF *Node*, even when there is an apparent name similarity. If the scheme is not "*cid*", the *Manager* SHALL keep the Comment available for the *Worker* to retrieve at least until the *Worker* completes or aborts the Job.
file:...	w←			r			URL whose scheme is "*file*".
http:...		w←			r		URL whose scheme is "*http*".
https:...		w?			r?		URL whose scheme is "*https*".

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
cid:...		w←			r		URL whose scheme is "*cid*".
all remaining values	!w			r?			
DescriptiveName	w←			r?			SHALL occur in ***Root Node***, indicating a single line Job Title; SHOULD occur in ***Subnodes*** with other values. If the ***Worker*** identifies the ***Node*** to an operator, it SHOULD include the *@DescriptiveName* in any such identification. **Note**: many ***Devices*** have limited possibilities to display the Job description. The string value SHOULD be as short as possible.
ICSVersions	w←			r?			SHALL occur in ***Root Node***, MAY occur in ***Subnodes***. Each NMTOKEN value has the syntax: <ICSShortName>_L <ICSLevelNumber>-<MajorVersion>. <MinorVersion> The ***Manager*** SHALL supply a set of NMTOKEN values, one for each ***ICS*** with which the ***JDF Instance*** complies. The NMTOKEN values are unordered. Each ***ICS*** specifies only the unique NMTOKEN value(s) that pertain to its domain, even if the ***ICS*** requires other ***ICSs*** to be Supported.
Base_L0-1.5	w?			r?			Specifies that the JDF ***Node*** conforms to [Base-ICS] level 0. SHALL only be specified if unidirectional JDF is acceptable to the ***Manager***.
Base_L1-1.5	w			r?			Specifies that the JDF ***Node*** conforms to [Base-ICS] level 1.
Base_L2-1.5		w		r?			Specifies that the JDF ***Node*** conforms to [Base-ICS] level 2.
all remaining values	w←			r?			Values specified in other ***ICSs***.
ID	w r?			r? w←			SHALL be supplied if a new ***Node*** is created. SHALL NOT be modified.
JobID	w←			r			SHALL occur in ***Root Node***, MAY occur in ***Subnodes***. **r-Test:** ***Worker*** SHALL preserve *@JobID* values, and SHALL use this value when sending Messages that require the ***Node***'s *@JobID* and SHALL use this value to identify Jobs that are specified in received Messages.

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
JobPartID	w r?			r w←			Each JDF ***Node*** (Product, Process Group, Combined Process, and Process) SHALL have a *@JobPartID* and its value SHALL be unique within the context of all ***JDF Instances*** that have the same *@JobID* in the print shop's workflow. When creating a JDF ***Subnode***, a ***Worker*** SHALL generate the new *@JobPartID* by adding a suffix to the parent JDF ***Node***'s *@JobPartID*. Each suffix SHALL start with a period "." and SHALL NOT exceed 3 characters including the *period*. The resulting *@JobPartID* SHALL NOT exceed 63 characters. **Note:** *@JobPartID* is required even at the root level. **r-Test:** ***Worker*** SHALL preserve *@JobPartID* values and SHALL use this value when sending Messages that require the ***Node***'s *@JobPartID* and SHALL use this value to identify ***Nodes*** that are specified in received Messages.
MaxVersion	w←			r			SHALL be in ***Root Node***, MAY be in ***Subnodes***. **r-Test:** Returned JDF ***Node*** contains no Elements or Attributes from newer versions of JDF than specified version.
1.5	w			r			A value higher than 1.5 MAY be specified.
Status	w r?			r w			See [JDF1.5] Sections 3.2 "JDF Node" and 4.3 "Execution Model", and this document (Base ICS) Section A.1 "*Interfaces for Conformance Requirements*" of this document. **r-Test:** A ***Worker*** SHALL NOT execute Nodes whose status is "*Completed*" or "*Aborted*".
all values	w← r?			r w←			
Type	w			r?			As specified in [JDF1.5]. Domain ICS's specify stricter read requirements if necessary.
Version	w← r?			r? w?			SHALL be in ***Root Node***, MAY be in ***Subnodes***. See [JDF1.5] Section 3.13 "JDF Versioning" for information about versioning.
1.5	w r?			r? w			
xmlns	w←			r?			SHALL be in ***Root Node***, MAY be in ***Subnodes***. The namespace for JDF MAY be the default namespace or any prefixed namespace.

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
http://www.CIP4.org/JDFSchema_1_1	w			r?			**Note:** that for all 1.x versions of [JDF1.5], the namespace URI is the same.
xmlns:xsi	w←			r?			SHALL be in ***Root Node***, MAY be in ***Subnodes***.
http://www.w3.org/2001/XMLSchema-instance	w			r?			
xsi:type	w			r?			Helps JDF Schema aware implementations to identify specific ***Node*** types.
AncestorPool	w?			r?			SHALL only be specified in the ***Root Node*** of a spawned JDF. See Table 14: AncestorPool.
AuditPool	w r?			r? w			See Table 11: AuditPool.
JDF	w?			r			Child JDF ***Node***(s). See Table 7: JDF Node.
ResourcePool	w←			r			See Table 8: ResourcePool.

6.2 ResourcePool

Note that ***Conformance Requirements*** for the ResourcePool, while appearing in the same table, are not meant to imply that all Resources shown in the table appear in the local JDF ***Node***'s ResourcePool, or even in the same ResourcePool. Resources appearing in ResourcePool ***Conformance Tables*** in any ***ICS*** are only stating that the Resources SHALL appear in some ResourcePool, and are linked to one or more ***Nodes***. See [JDF1.5] Sections 3.8 "ResourcePool and its Resource Children" and 3.9 "ResourceLinkPool and ResourceLink".

Table 8: ResourcePool

From: [JDF1.5] Table 3-9

Referenced by: JDF Node

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Resource	w←			r←			Abstract Resource. See Table 9: Abstract Resource.

6.3 Abstract Resource

Table 9: Abstract Resource

From: [JDF1.5] Table 3-10

Referenced by: ResourcePool

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Class	w			r			Required in [JDF1.5].
ID	w			r			Required in [JDF1.5].
Status	w← r←			r← w←			Managers SHALL write appropriate values for the *@Status* Attribute of all Resources. Workers SHALL update the *@Status* Attribute of all Resources appropriately. For example, output Resources would normally be updated. See [JDF1.5], table 3-10 in Section 3.8.3 "Abstract Resource" for possible values and the definitions of how they are to be used.

6.4 List of Resources

Table 10: List of Resources

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Device	w←			r			If it is desired to target execution of a ***Process*** to be done by a specific ***Device***, **Device** SHALL be specified. See Table 22: Device.
NodeInfo	w← r			r w←			If different status values are necessary for different Partitions of the ***Node***, the **NodeInfo** Resource SHALL be written. The same **NodeInfo** Resource SHALL NOT be linked from more than one Node. It MAY be referenced multiple times from Ancestor Elements. See Table 23: NodeInfo.

6.5 AuditPool

This section specifies the AuditPool. See other ***ICSs***, such as the [MIS-ICS], for additional AuditPool requirements.

Table 11: AuditPool
From: [JDF1.5] Table 3-29
Referenced by: JDF Node

Name or Value	*Manager*			*Worker*			Description
Level ➔	1	2	3	1	2	3	
Audit	w←			r?			See Table 12: Abstract Audit.

6.6 Abstract Audit

Table 12: Abstract Audit
From: [JDF1.5] Table 3-30
Referenced by: AuditPool

Name or Value	*Manager*			*Worker*			Description
Level ➔	1	2	3	1	2	3	
AgentName	w r?			r? w			
AgentVersion	w r?			r? w			
ID	w r?			r? w			
TimeStamp	w r?			r? w			

6.7 List of Audit Elements

This section specifies the Audit Elements to Support.

Table 13: List of Audit Elements
From: [JDF1.5] Table 3-31

Name or Value	*Manager*			*Worker*			Description
Level ➔	1	2	3	1	2	3	
Created	w r?			r? w←			Created Audit Elements have no required ***Traits*** other than those defined in Abstract Audit. If the ***Worker*** creates the ***Node***, it SHALL write the Created Audit Element. See [JDF1.5].
Modified	w?			r?			SHOULD supply if significantly modifying the

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
	r?			w?			JDF. Modified Audit Elements have no required *Traits* other than those defined in Abstract Audit. See [JDF1.5].

6.8 AncestorPool

The information about ancestor JDF *Nodes* in a spawned Job SHALL be specified in the AncestorPool/Ancestor Elements. For details see [JDF1.5] Section 3.4 "AncestorPool". A conforming reader SHALL be capable of extracting information from the AncestorPool.

Table 14: AncestorPool

From: [JDF1.5] Table 3-7

Referenced by: JDF Node

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Ancestor	w			r?			See Table 15: Ancestor.
Part	w←			r?			The Part Elements contain the Partition Attributes that are filled in *@Status* Messages, PartStatus Elements, Audit Elements, etc. See [JDF1.5].

6.9 Ancestor

Table 15: Ancestor

From: [JDF1.5] Table 3-8

Referenced by: AncestorPool

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
All Attributes defined in Table 7: JDF Node.	w←			r?			The *Conformance Requirements* that are defined in Table 7: JDF Node also apply to the same Attributes and their values in Ancestor with the exception of the *@ID* Attribute (see next row).
ID	!w			r?			
NodeID	w			r?			SHALL be a copy of JDF/@ID for the *Node* that this Ancestor represents.

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
CustomerInfo	w←			r?			If the JDF *Node* that this Ancestor Element represents has a linked **CustomerInfo** Resource, the **CustomerInfo** refelement SHALL be specified. See [JDF1.5].
NodeInfo	w←			r?			If the JDF *Node* that this Ancestor Element represents has a linked **NodeInfo** Resource, the **NodeInfo** refelement SHALL be specified. See Table 23: NodeInfo.

7 Conformance Tables – Abstract Resources

7.1 Abstract Partitionable Resource

This section defines general ***Conformance Requirements*** for Resources. It specifically defines the ***Conformance Requirements*** pertaining to Partitioning of Resources.

Any conforming ***Consumer*** of JDF SHALL ***Support*** reading of Partitioned Resources and behave in a manner that is specified in [JDF1.5] Section 3.10.6 “PartIDKeys Attribute and Partition Keys”.

Table 16: Abstract Partitionable Resource

From: [JDF1.5] Table 3-25

Superclass of: Component, Device, NodeInfo

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Class	w			r			Required in [JDF1.5].
ID	w r?			r w←			SHALL be supplied if a new Resource is created. **r-Test:** SHALL NOT be modified.
PartIDKeys	w← r?			r? w←			Partitioned Resources SHALL contain a *@PartIDKeys* Attribute with values as specified in [JDF1.5] Section 3.10.6 “PartIDKeys Attribute and Partition Keys”. Other ***ICSs*** may restrict the list of allowed values of *@PartIDKeys* for a given Resource, and specify stricter read conformance if necessary.
PartUsage	w? r			r w?			**r-Test:** Domain ICS’s specify read conformance tests.
Implicit	w? r			r w?			**r-Test:** Domain ICS’s specify read conformance tests.

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Explicit	w? r			r w?			**r-Test:** Domain ICS's specify read conformance tests.
Sparse	w? r?			r? w?			Domain ICS's may have higher conformance requirements for *Sparse* Partitioning.
Status	w r			r w←			A ***Worker*** SHALL update the *@Status* of any Resource that it produces as output to "*Available*". A ***Worker*** SHOULD update the *@Status* of any Resource that it consumes completely as input to "*Unavailable*". **r-Test:** ***Worker*** SHALL NOT execute ***Nodes*** that have input Resources where the Resource's *@Status* Attribute has a value "lower" than the value specified in *@MinStatus*, as defined in [JDF1.5] Table 3-10 in Section 3.8.3 "Abstract Resource".

7.2 ResourceLink and ResourceLink/AmountPool/PartAmount

Table 17: ResourceLink and ResourceLink/AmountPool/PartAmount

From: [JDF1.5] Table 3-16 and Table 3-19

Referenced by: AmountPool (for ResourceLink/AmountPool/PartAmount)

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
ActualAmount	w← r?			r w?			PhysicalLink/@ActualAmount SHALL NOT be written if AmountPool exists. If @Usage="Input", the amount that is actually consumed. If @Usage="Output", the amount that is actually produced. The ***Manager*** SHALL write the *@ActualAmount* if the Resource has already been partially produced before. A ***Worker*** SHOULD update the *@ActualAmount* of any **PhysicalResource** that it produces or consumes. See [JDF1.5] Section 3.10.4 "Resource Amount" for details. Domain ICS's may specify additional requirements for updating *@ActualAmount* for any specific kind of **PhysicalResource**. **r-Test:** ***Worker*** SHALL take this value into

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							consideration when determining quantity to be produced (@Amount - @ActualAmount) or when updating production or consumption (existing *@ActualAmount* value SHALL be increased, not overwritten).

Amount	w←			r←			PhysicalLink/@Amount SHALL NOT be written if AmountPool exists.
							SHALL NOT be specified if Part/@Condition is other than "*Good*".
							If (@Usage="Input"): The amount that is planned to be consumed. The amount to be consumed is (@Amount - @ActualAmount).
							If (@Usage="Output"): The amount that is planned to be produced. The amount to be produced is (@Amount-@ActualAmount). See [JDF1.5] Section 3.11.4.1 "Evaluating and Updating Amount-Related Attributes in a Device".
							A ***Manager*** MAY supply the amount that is expected to be consumed by the ***Process***. A ***Manager*** SHALL specify the amount of the output Resource to be produced. See [JDF1.5] Section 3.11.4 "Resource Amount" for details.
							r-Test: ***Worker*** SHALL read *@Amount* on output ResourceLink Elements in order to calculate the correct amount to be consumed or produced.

MaxAmount	w?			r?			PhysicalLink/@MaxAmount SHALL NOT be written if AmountPool exists. If @Usage="Input", and @Condition="Waste", this figure specifies the maximum amount of input that is estimated to be wasted. If @Usage="Output", and @Condition="Good", this figure specifies the maximum amount that can be produced (Overage is (@MaxAmount minus @Amount)). Domain ICS's may specify additional requirements for reading *@MaxAmount* for specific kinds of Physical Resources.
MinAmount	w?			r?			PhysicalLink/@MinAmount SHALL NOT be written if AmountPool exists. If @Usage="Output" and @Condition="Good", this figure specifies the minimum amount that can be produced

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							(Underage is (@Amount minus @MinAmount)). Domain ICS's may specify additional requirements for reading *@MinAmount* for specific kinds of Physical Resources.
MinStatus	w? r?			r w←			If a ***Worker*** adds a new ResourceLink, it SHALL write this value. **r-Test:** ***Worker*** SHALL NOT execute ***Nodes*** that have input Resources where the Resource's *@Status* Attribute has a value "lower" than the value specified in *@MinStatus*, as defined in [JDF1.5] Table 3-10 in Section 3.8.3 "Abstract Resource".
ProcessUsage	w← r			r w←			If multiple Resources of the same type are used by a ***Process***, *@ProcessUsage* SHALL be used to distinguish them as defined in [JDF1.5] Chapter 6 "Processes". **r-Test:** The ***Manager*** and ***Worker*** SHALL conform to read requirements for the linked Resource as specified in other ICS's.
rRef	w r			r w←			If a ***Worker*** adds a new ResourceLink, it SHALL write this value. This Attribute SHALL reference a Resource that is a direct child of a ResourcePool. **r-Test:** The ***Manager*** and ***Worker*** SHALL conform to read requirements for the linked Resource as specified in other ICS's.
Usage	w r			r w←			If a ***Worker*** adds a new ResourceLink, it SHALL write this value. **r-Test:** The ***Manager*** and ***Worker*** SHALL conform to read requirements for the linked Resources as specified in other ICS's, and SHALL update Output ResourceLink Elements as specified in other ICS's. See [JDF1.5].
AmountPool	w← r?			r w?			SHALL NOT be specified in PartAmount. ***Managers*** MAY create placeholders for the ***Worker*** to place actual values in. ***Worker*** MAY create the AmountPool, if it does not exist, before it returns the ticket to the ***Manager***. If the ***Worker*** creates the AmountPool, the ***Worker*** SHALL move the *@ActualAmount*, *@Amount*, *@MaxAmount* and *@MinAmount* Attributes to the AmountPool

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							and assume that amounts on the Link need to go into the Good Partition. See Table 18: AmountPool.
Part	w← r?			r w←			If the Part Element is part of the PartAmount Element, Part SHALL be specified. A ***Worker*** SHALL read and Support ResourceLink Elements that reference one or more Partitions of a Resource. **r-Test:** The ***Worker*** SHALL conform to read requirements for the linked Resource as specified in other ICS's. See [JDF1.5]. See Table 19: Part.

7.2.1 AmountPool

Table 18: AmountPool

From: [JDF1.5] Table 3-18

Referenced by: ResourceLink

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
PartAmount	w			r			See Table 17 (as ResourceLink/AmountPool/PartAmount).

7.2.2 Part

Table 19: Part

From: [JDF1.5] Table 3-26

Referenced by: ResourceLink and ResourceLink/AmountPool/PartAmount

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Condition	w← r?			r w←			Partition key identifying the condition of the Partition.
Good	w← r?			r w			On an Output ResourceLink, this Partition identifies the Good quantity that is produced by the current ***Process***, and will be used by the next ***Process***, if there is one.

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
							On an Input ResourceLink, this Partition identifies the Good quantity that is expected to be available to the current ***Process***. **r-Test:** The ***Worker*** SHALL use this Partition to determine the target amount to be produced.
Waste	w? r?			r? w?			On an Output ResourceLink, this Partition identifies the quantity that is not Good and cannot be used by the next ***Process***. Input ResourceLink Elements SHOULD NOT specify a @Condition="Waste" Partition.
Reusable	w? r?			r? w?			On an Output ResourceLink, this Partition identifies the quantity that is not Good, but can be used by the next Process for setup. On an Input ResourceLink, this Partition identifies the quantity that is not Good, but can be used by the current Process for setup.

7.3 Abstract ResourceRef

A ResourceRef Element may occur where [JDF1.5] specifies a data type of refelement. Conforming ***Consumers*** SHALL ***Support*** reading ResourceRef Elements wherever a data type of refelement is specified in [JDF1.5].

Table 20: Abstract ResourceRef

From: [JDF1.5] Table 3-22

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
rRef	w			r			This Attribute SHALL NOT reference a Resource that is a ***Subelement*** of another Resource. In a JDF the referenced Resource SHALL be a direct child of a ResourcePool. **r-Test:** The read requirements of the referenced Resource SHALL be met.
Part	w?			r			A ***Worker*** SHALL read and ***Support*** References to a Partition of a Resource. **r-Test:** The read requirements of the referenced Resource SHALL be met. See [JDF1.5].

8 Conformance Tables – Resources

8.1 Component

Table 21: Component

From: [JDF1.5] Table 9-4

Subclass of: Abstract Partitionable Resource

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Condition	w←			r			Partition key describing the condition of the Resource. If all Partitions use the same physical Resource the Resource does not need to be Partitioned. The AmountPool in the Link specifies the amounts per Partition.

8.2 Device

This section defines general ***Conformance Requirements*** for the **Device** (Implementation) Resource, which a ***Manager*** MAY supply in any Combined, Process or Process Group ***Node***. See section 9.1 "Complete Job Instance versus a (spawned) Process Node", [JDF1.5] Section 3.8.5.3 "Implementation Resource", [JDF1.5] Section 7.2.58 "Device", and [JDF1.5] Section 4.2.1 "Determining Executable Nodes").

Table 22: Device

From: [JDF1.5] Table 9-6

Subclass of: Abstract Partitionable Resource

In: List of Resources

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
DeviceID	w?			r			**r-Test:** If the *@DeviceID* specified does not match the ***Worker***'s *@DeviceID*, the ***Worker*** SHALL NOT execute the ***Node***.

8.3 NodeInfo

Table 23: NodeInfo
From: [JDF1.5] Table 8-165
Subclass of: Abstract Partitionable Resource
Referenced by: Ancestor
In: List of Resources

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
DescriptiveName	w?			r?			If the Worker identifies individual work steps to an operator, it SHOULD include the *@DescriptiveName* of the appropriate Partition in any such identification. What constitutes a "work step" is left to be defined within domain ICSs.
NodeStatus	w← r?			r← w←			SHALL be written and evaluated if and only if JDF/@Status = "*Part*". **r-Test:** ***Workers*** SHALL NOT execute Node Partitions whose status is "*Completed*" or "*Aborted*".
TargetRoute	w?			r			Where the ***Worker*** returns or forwards the entire JDF Instance after processing as much of the JDF as it can. MAY appear only within the Root Element of the **NodeInfo** Resource, and SHALL NOT appear in any lower levels or in Partition leaves. If *@TargetRoute* is not supplied, the ***Worker*** SHALL send the completed JDF according to [JDF1.5] *@TargetRoute* description. A ***Worker*** that only complies to Level 0 of this ***ICS*** NEED NOT ***Support*** *@TargetRoute*. **r-Test:** Upon completion of processing, the ***Worker*** SHALL place the entire JDF Instance at the location identified by *@TargetRoute*.
file:...	w←			r			URL whose scheme is "*file*".
all remaining values	!w			r?			

9 Conformance Rules – Job Submission

Manager/***Worker*** interfaces which involve JDF Job submission SHALL conform to the requirements below. ICS documents which inherit from this ICS, but do not involve JDF Job submission, NEED NOT conform to the requirements below.

See Section 2 Glossary for definitions of ***JDF Instance File***, ***JDF MIME File*** and ***Referenced File***, all of which appear in the subsections below with hot links on the first occurrence.

9.1 Complete Job Instance versus a (spawned) Process Node

When creating a JDF, the ***Manager*** has two options:

- Create a complete JDF Instance describing the complete production process for the Job; or
- Create multiple JDF Instances, one for each ***Worker***, with only the information required by that ***Worker***. A ***Manager*** can create this type of JDF Instance by spawning off the individual Process Node(s) that the ***Worker*** requires.

To be conformant to this ICS, the ***Worker*** SHALL be able to read and process both kinds of JDF Instances. If a ***Worker*** receives a complete JDF Instance, it SHALL be able execute the JDF Instance. For details see [JDF1.5] Section 4.2.1 "Determining Executable Nodes".

A JDF Instance MAY include more than one Process Node for a target Device. To be conformant to this ICS, a ***Worker*** SHALL be able to execute multiple Process Nodes from a JDF Instance. For details see [JDF1.5] Section 4.2.1 "Determining Executable Nodes" where (DeviceCap/@ExecutionPolicy = "AllFound"). The ***Manager*** MAY target a JDF Node to a specific Device by linking a Device Resource to the JDF Node (see Section 8.2 "Device").

9.2 Hot Folders

When a ***Manager*** submits a JDF Instance to a ***Worker***, the ***Manager*** places the JDF Instance directly into the ***Hot Folder*** of the Device. To be compliant to Level 1 of this ICS, both the ***Manager*** and the Device ***Worker*** SHALL Support JDF ***Hot Folders***. Note: starting with Version 1.4 of JMF ICS, ***Hot Folders*** are OPTIONAL at level 2, while they remain REQUIRED for Level 1.

9.2.1 Requirements for Producers

A level 1 ***Producer*** (***Manager*** or ***Worker***) SHALL be capable of writing a ***JDF Instance File*** and a level 2 ***Manager*** MAY be capable of writing a ***JDF MIME File*** into a ***Hot Folder***.

If file *F* contains references to other files f_1, … f_n (called ***Referenced Files***) and the receiving ***Consumer Supports Referenced Files***, there are two types of ***Referenced Files*** to ***Support***.

- The ***Producer*** intends that some files be accessed in a location outside the ***Hot Folder***. This specification does not address this case.
- The ***Producer*** intends that some files f_1, … f_n be accessed in a location inside the ***Hot Folder***. The ***Producer*** SHALL NOT write such files directly into the ***Hot Folder***. Instead, the ***Producer*** SHALL ***Support*** both of the following mechanisms:

    1. **Create a directory D:** The ***Producer*** creates a directory *D* within the ***Hot Folder*** and writes the ***Referenced Files*** f_1, … f_n into the directory *D* or its sub-directories, and
    2. **Write into an existing directory:** Write the ***Referenced Files*** f_1, … f_n into an existing directory within the ***Hot Folder*** in an implementation dependent manner.

Note: a single ***JDF Instance File*** or ***JDF MIME File*** could contain references to both types of ***Referenced Files***.

Note: a ***Consumer*** typically ***Supports*** only one of the above two mechanisms for writing ***Referenced Files*** into a ***Hot Folder***. This document does not specify how a ***Producer*** determines the mechanism that a ***Consumer Supports***, or how the ***Producer*** determines the name of the sub-directory (for either case).

A ***Producer*** SHALL NOT remove any files or directories that are accessible via the ***Hot Folder***, including those files and directories that the ***Producer*** writes. Furthermore, the ***Producer*** SHALL NOT submit multiple Jobs that reference the same files in sub-directories of a ***Hot Folder***.

9.2.2 Requirements for Consumers

A level 1 ***Consumer*** SHALL be able to read and process any JDF file in a ***Hot Folder*** under its control.

The ***Consumer***'s ***Hot Folder*** SHALL be configured for a ***Producer*** to write a ***JDF Instance File*** into the ***Hot Folder***.

If a ***Consumer Supports Referenced Files***, the ***Consumer***'s ***Hot Folder*** MAY accept a ***JDF MIME File***.

Level 1 ***Consumers*** SHALL ***Support*** at least one of the options below that corresponds to one of the options in section 9.2.1 Requirements for Producers:

3. **Create a directory D:** the ***Hot Folder*** SHALL allow ***Producers*** to create directories within the ***Hot Folder***.
4. **Write into an existing directory:** the special directory for ***Referenced Files*** within the ***Hot Folder*** SHALL allow ***Producers*** to write files.

If a ***Consumer Supports Processes*** that require ***Referenced Files***, the ***Consumer*** SHALL ***Support*** the URLs in Table 24 for accessing ***Referenced Files*** from a ***JDF Instance*** in any file, whether the file is a ***JDF Instance File*** or a ***JDF MIME File***.

Note: A ***Consumer*** or some companion software typically removes ***JDF Instance Files*** and ***JDF MIME Files*** along with any ***Referenced Files*** and directories from the ***Hot Folder*** at some time after the ***Consumer*** no longer needs them. However, this ***ICS*** specifies no requirements for such file and directory removal, and an implementation may or may not use any housekeeping mechanism that it chooses. For example, an implementation may choose to do no housekeeping.

Table 24: Consumer Supported Schemes

Name or Value	Manager			Worker			Description
Level ➔	1	2	3	1	2	3	
Any Attribute whose Data Type is "URL".	w← r?			r← w←			If the scheme is not "*cid*", the ***Producer*** SHALL keep the ***Referenced File*** available for the ***Consumer*** to retrieve at least until the ***Consumer*** finishes processing. **r-Test:** As specified in other ICS documents for specific instances of Attributes of Data Type URL.
file:…	w← r?			r w?			URL whose scheme is "*file*".
./…	w← r?			r w?			Any Relative URL relative to the ***Hot Folder***. Relative URLs SHALL be ***Supported*** by ***Workers*** assuming the "*file*" scheme and MAY be ***Supported*** assuming other schemes. See [FileURL-AN].
http:...		w← r?			r w?		URL whose scheme is "*http*".
cid:…		w← r?			r w?		URL whose scheme is "*cid*".

10 References

10.1 Normative References

[ISO8601:2004] Data elements and interchange formats - Information interchange - Representation of dates and times, published 2004, available at http://www.iso.ch/iso/en/prods-services/ISOstore/store.html.

[JDF1.5] JDF Specification, Version 1.5, published December 31, 2013. Available at: http://www.cip4.org.

[JDF-logo] CIP4 Guidelines: *For CIP4 and JDF Logo Use*, April 14, 2004, to be updated to include ***ICS*** conformance. Available at: http://www.cip4.org.

[RFC2387] Levinson, E., The MIME Multipart/Related Content-type, RFC 2387, August 1998.

[RFC2392] Levinson, E., Content-ID and Message-ID Uniform Resource Locators, RFC 2392, August 1998.

[RFC3986] Berners-Lee, T., Fielding, R. and L. Masinter, Uniform Resource Identifier (URI): Generic Syntax, RFC 3986, January 2005. Updated by RFC 6874.

[RFC3987] M. Duerst and M. Suignard, *Internationalized Resource Identifiers (*IRIs*)*, RFC 3987, January 2005.

[RFC6874] Carpenter, B., Cheshire, S. and Hinden, R., *Representing IPv6 Zone Identifiers in Address Literals and Uniform Resource Identifiers*, RFC 6874 updates RFC3986. February 2013.

[XPath] Clark, James and Steve DeRose, XML Path Language (XPath) http://www.w3.org/TR/1999/REC-xpath-19991116, November 16 1999.

[XMLSchema] Peterson, David, et al., W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes, http://www.w3.org/TR/xmlschema11-2/, April 5 2012.

10.2 Informative References

[FileURL-AN] CIP4 Application Note: Use of the File URL in JDF, published 12 November 2003. Available at: http://www.cip4.org.

[MIS-ICS] MIS ICS, Version 1.5, published December 2009. Available at http://www.cip4.org.

Appendix A: How to Read ICS Documents

This appendix explains how to read ***ICS*** documents and defines the notation used in the ***Conformance Tables*** of all ***ICS*** documents.

Need for Multiple ICSs

Although a single ***ICS*** that encompasses all JDF-enabled Products might appear to be the best solution, such an ***ICS*** would burden JDF-enabled Products with the necessity of implementing many unneeded ***Conformance Requirements***. For example, a ***Stitching Process*** for JDF-enabled Products in the postpress sector would add unneeded complexity to JDF-enabled Products in the conventional printing sector. Hence, there is a need for a separate ***ICS*** for each Product sector; Product sectors include prepress, digital printing, conventional printing and postpress. Each such ***ICS*** defines a subset of JDF for all JDF-enabled Products in its Product sector.

Avoiding Replication of Conformance Requirements

If ***Conformance Requirements*** for each Product sector were described by a single ***ICS***, some ***Conformance Requirements*** would be replicated in many of the ***Product-Sector ICSs***. To avoid replication, common ***Conformance Requirements*** can be factored out and placed into another ***ICS***. In particular, ***Conformance Requirements*** that are common to all ***Product-Sector ICSs*** have been factored out and placed in a separate ***ICS*** – the [Base-ICS] (this document). Common ***Conformance Requirements*** that are ***MIS*** in nature and are not in the [Base-ICS] have been factored out from the ***Product-Sector ICSs*** and placed in a separate ***ICS*** – the [MIS-ICS].

Conformance Requirements that are in common with any ***ICS*** and the [JDF1.5] have been factored out from the ***ICS***. Here are some examples:

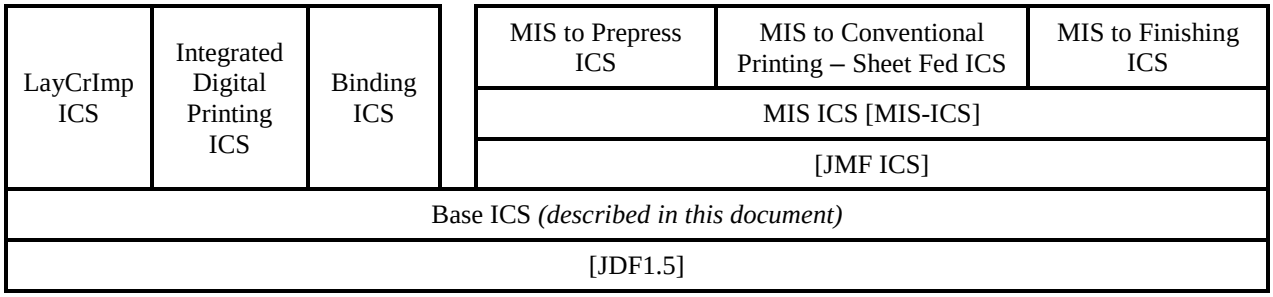
The [JDF1.5] specifies the maximum value for an integer type but does not specify the maximum length of a string. No ***ICS*** replicates such information about an integer. However, the [Base-ICS] does specify the maximum length of a string.

If an ***ICS*** specifies ***Support*** for an ***Element*** and the [JDF1.5] specifies that some of its Attributes are optional, the ***ICS*** is silent about those Attributes that remain optional in the ***ICS*** to avoid replicating ***Conformance Requirements*** of the [JDF1.5].

Some Product sectors have a wide range of JDF-enabled Products, from simple to complex. To accommodate this range of capabilities, there is a need for multiple sets of ***Conformance Requirements***. There could be a separate ***ICS*** for each set of ***Conformance Requirements***. Instead, there is a single ***ICS*** that specifies multiple sets of ***Conformance Requirements***. Each set is called a ***Conformance Level***. ***Conformance Level*** 1 specifies ***Conformance Requirements*** for the simplest JDF-enabled Products. ***Conformance Level*** 2 specifies ***Conformance Requirements*** for more complex JDF-enabled Products; the ***Conformance Requirements*** are a superset of the Level-1 ***Conformance Requirements***.

The [JDF1.5], the [Base-ICS], the [MIS-ICS] and all the ***Product-Sector ICSs*** with their levels form a relationship that is similar to a protocol stack as shown in Figure 1.

Figure 1: Example of an ICS Stack



Ramifications for Implementers

Although common ***Conformance Requirements*** have been factored out into separate ICS-documents, ***Product-Sector ICSs*** will still contain these common ***Conformance Requirements***. In the ***Product-Sector ICSs*** these common ***Conformance Requirements*** are color coded to indicate they are common and have been copied from an ICS document higher in the ICS Stack.

As a result, ICSs are purposefully redundant and an implementer of a JDF-enabled Product only needs to consult a single document in order to get a complete picture.

This is a departure from pre-1.5 ICS-documents that were purposefully not redundant. An implementer of such a JDF-enabled Product had to consult the following documents in order to get a complete picture:

- The appropriate ***Product-Sector ICS***.
- All ***ICSs*** referenced by the ***Product-Sector ICS***, including the [Base-ICS] and possibly the [MIS-ICS].
- The parts of the [JDF1.5] referenced from a relevant ***ICS***, such as definitions of ***Elements*** and data types.

For example, if an implementer wanted to know what parts of the **Component** Resource to implement for a Binding Product, the implementer would need to read all information in the following places:

Binding ICS sections: **Component**, ComponentLink – Input and ComponentLink – Output.
MIS ICS sections: **Component** – Output and ComponentLink – Output.
Base ICS sections: Resource and ResourceLink.
[JDF1.5] **sections:** **Component**, **PhysicalResource**, Resource, ComponentLink, PhysicalLink and ResourceLink.

Starting with version 1.5, a single ICS document provides all the required information.

A.1 Interfaces for Conformance Requirements

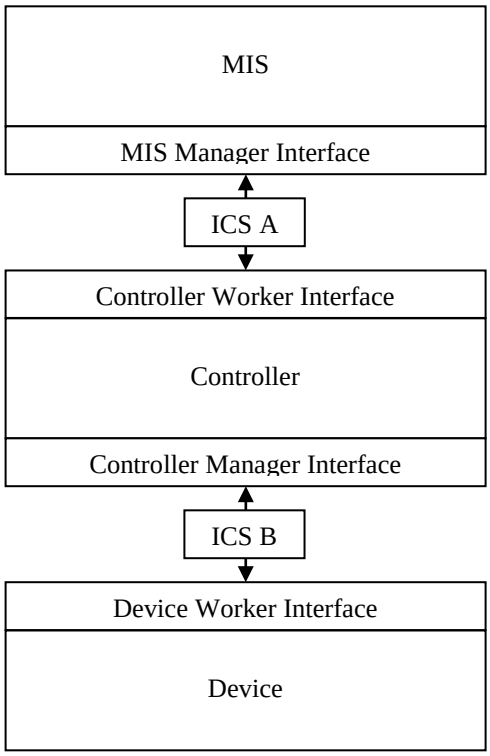
The [JDF1.5] defines a hierarchical relationship of workflow roles, which include the ***MIS***, ***Controllers*** and ***Devices***. See [JDF1.5] Figure 2.1. A ***Device*** is at the bottom of the hierarchy. It executes a ***Process*** and may send information back up the hierarchy about the results of executing the ***Process***. A ***Controller*** is somewhere above the bottom of the hierarchy. It causes some ***Device*** or other ***Controller*** to execute a ***Process***, and may receive information back from the ***Device*** or other ***Controller*** about the executed ***Process***. It may then send this information back up the hierarchy. The ***MIS*** is at the top of this hierarchy and acts as the master ***Controller***. Each workflow component in this hierarchy is connected to other workflow components, possibly via the network.

The [JDF1.5] defines an additional workflow role called an ***Agent***. An ***Agent*** can write JDF. It can create JDF ***Nodes*** or it can modify existing JDF ***Nodes***. In most cases, the software that implements an ***MIS***, ***Controller*** or ***Device*** also implements an ***Agent***.

In the hierarchical model in Figure 2: Manager - Worker Interfaces, a ***Controller*** deals with workflow components above it and below it. An ***MIS*** deals only with workflow components below it and a ***Device*** deals only with workflow components above it. This observation suggests an interface for dealing with workflow components above, and another interface for dealing with workflow components below. In this document, the term ***Manager Interface*** describes the interface that deals with workflow components below, and the term ***Worker Interface*** describes the interface that deals with workflow components above. In [JDF1.5] Figure 2-1 “Example of JDF and JMF workflow interactions”, wherever an arrow comes from below, there is a ***Manager Interface*** and wherever an arrow comes from above, there is a ***Worker Interface***.

A ***Controller***/***Agent*** component SHALL implement both the ***Manager Interface*** and the ***Worker Interface***. A ***Device***/***Agent*** component SHALL implement the ***Worker Interface***, but not the ***Manager Interface***. An ***MIS*** SHALL implement the ***Manager Interface***.

Figure 2: Manager - Worker Interfaces



A.2 Content of Conformance Tables

Section 6 “Conformance Tables” contains a series of ***Conformance Tables*** that specify the ***Conformance Requirements*** for various ***Elements*** and their ***Traits***. This section describes the content and format of ***Conformance Tables***. Other ***ICSs*** have similar ***Conformance Tables*** for ***Elements***. In addition some ***ICSs*** have ***Conformance Tables*** for Input and Output Resources of ***Processes***.

Each ***Conformance Table*** describes the ***Conformance Requirements*** for a single ***Element*** of a ***JDF Instance*** or ***JMF Message***. Each row of a ***Conformance Table*** contains a single ***Trait*** of the ***Element*** and defines the ***Conformance Requirements*** for ***Managers*** and ***Workers***, in order to comply to the levels defined in the ***ICS***.

When a table contains an ***Abstract Element***, another table lists the ***Conformance Requirement*** for (some of) its ***Derived Elements***. The ***Conformance Requirements*** of each ***Derived Element*** in such a table are similar to those of ***Conformance Tables*** for ***Elements***.

Note: the absence of a ***Trait*** from an ***ICS*** means that the ***Trait*** is optional, according to that ***ICS***. However, other ***ICSs*** may require the ***Trait***. If [JDF1.5] specifies that X is optional and all lower ***ICSs*** either are silent about X or specify that X is optional. The “?” in the Name Column in [JDF1.5] generally means that the ***Trait*** is optional. However, the Description column may state that a ***Trait*** is conditionally required. For such an example, see ResourceLink/@ProcessUsage in [JDF1.5].

The ***Conformance Requirement*** for any ***Trait*** is always conditioned by the ***Conformance Requirement*** for its containing ***Element***. For example, if some Attribute is required, but the containing ***Element*** is optional, the entire ***Element*** may be omitted.

A.2.1 Format of Conformance Tables

The format of each ***Conformance Table*** is similar to the tables for ***Elements*** in the [JDF1.5]. In the [JDF1.5], most ***Element*** tables have three columns: ***Name***, ***Data Type*** and ***Description*** with ***Attribute Values*** in the ***Description*** Column. The ***ICS Conformance Tables*** for ***Elements*** have no ***Data Type*** column and combine the names and values into a ***Name or Value*** column. In addition, the ***ICS Conformance Tables*** have two additional columns between the ***Name or Value*** and ***Description*** columns. Each of these two columns has one sub-column per level defined in the ***ICS***. Table 25 describes the format of ***Conformance Tables***.

Table 25: Format of Conformance Tables

Item	Column Name	Description
Table Caption		The caption specifies the ***Element*** or Process Resource described by the table. For example "Media" for an ***Element*** table or "Folding – Input Resources" for a ***Process*** Resources table.
Reference Line		One or more ***Reference Lines***, just below the Table Caption, have hot links to other tables that contain a reference to this table. Words in bold describe the relationship of the reference, such as **Referenced by** or **Derived from.**
1st column	Name or Value (the row has a white background)	For an ***Element*** table, when this column has a white background, it is like the *Name* column in the [JDF1.5]. It contains Attribute names, followed by ***Element*** names, each in alphabetic order.
1st column (alternative)	Name or Value (the row has a light gray background)	For an ***Element*** table, when the row has a light gray background, the cell in the ***Name or Value*** column contains either one ***Attribute Value*** or (for NMTOKENS and enumerations) one NMTOKEN or enumeration value. When a ***Conformance Table*** specifies the values of an Attribute, the values appear in a series of contiguous, light gray rows starting just below the row for the Attribute. Each value appears without quotes, as in the [JDF1.5]. If no ***Attribute Values*** are specified for a given Attribute, a conforming application SHALL ***Support*** at least one valid ***Attribute Value*** as defined by [JDF1.5] or a higher level ***ICS***. For NMTOKENS and enumerations, the [JDF1.5] and possibly the Attribute's Description cell specify the combinations of values (in rows) that comprise an NMTOKENS or enumerations ***Attribute Value***. There are two special ***Attribute Values***: ***all values*:** a shorthand for a series of rows that enumerate all values of an Attribute as specified by [JDF1.5]. Each row has one value as specified by the earlier paragraphs of this cell. ***all remaining values*:** identical in meaning to ***all values***, except that the Attribute has some explicitly specified values as well. This value is always the last value listed for an Attribute.
1st column (alternative)	Name (different column title)	For a ***Process*** Resources table, this column has Resource names.

Item	Column Name	Description
2nd column	Manager	Each of the sub-columns of this column is for a different ***Conformance Level***. Each sub-column specifies exactly two ***Conformance Requirements*** that apply to a conforming ***Manager*** – one for writing (see Table 26) and one for reading (see Table 27). The specified ***Trait*** or Resource is subject to these ***Conformance Requirements***. An ***ICS*** always specifies ***Conformance Requirements*** for Level 1.
3rd column	Worker	Replace the word "***Manager***" with the word "***Worker***" in the above cell.
2nd column (alternative)	***Producer***	**Note:** In some tables, the ***Conformance Table*** columns for ***Manager*** and ***Worker*** are relabeled ***Producer*** and ***Consumer*** (in italics). When a ***Manager*** writes a ***Trait***, it is a ***Producer*** and the ***Worker*** is the ***Consumer***. When a ***Worker*** writes a ***Trait***, it is the ***Producer*** and the ***Manager*** is the ***Consumer***.
3rd column (alternative)	***Consumer***	See above cell.
4th column	Description	If this column is blank, see [JDF1.5] for any conditions relating to the ***Trait***. The description of a ***Trait*** or Resource might also reference another ***Conformance Table***, or specify a condition for writing it.

A.2.2 Conformance Requirements for Writing and Reading

Two of the following three sections describe the reading and writing values that can appear in the six ***Manager*** and ***Worker*** sub-columns of a ***Conformance Table***. Table 26 in section A.2.2.2 shows values for writing a ***Trait*** or Resource. Table 27 in section A.2.2.3 shows values for reading a ***Trait*** or Resource. Section A.2.2.1 shows the meaning of a blank cell in one of these six sub-columns.

The ***Conformance Requirements*** are usually symmetric. For example, if a ***Conformance Requirement*** for writing a ***Trait*** applies to a ***Manager***, then there is usually a ***Conformance Requirement*** for reading a ***Trait*** that applies to a ***Worker***.

Note: in this section, ***Element E*** refers to the ***Element*** specified by the ***Conformance Table***'s caption, and ***Trait T*** in Element ***E*** is some ***Element***, ***Attribute*** or ***Attribute Value*** in ***Element E***'s ***Conformance Table***. ***Process P*** refers to the ***Process*** specified by the ***Conformance Table***'s caption and Resource ***R*** is some Resource in ***Process P***'s Input Resources ***Conformance Table*** or Output Resources ***Conformance Table***.

A.2.2.1 Blank Cell for a Conformance Level

If a ***Conformance Level***'s cell is blank, it inherits its ***Conformance Requirements*** from the cell to the immediate left, unless it is the level 1 cell. If the level 1 cell is blank, the ***Conformance Requirement*** defaults to "`w?r?`".

A.2.2.2 Conformance Requirements for Writing

All data written by ***Producers*** SHALL originate from data that exists in the relevant part of the application. Certification will include some amount of "randomness" to ensure this.

Table 26 lists the ***Conformance Requirements*** for writing in descending order of strength from "`w`" (the strongest) to "`!w`" (the weakest). Table 26 contains the following columns:

Value: the value that can appear in any of the six ***Manager*** and ***Worker*** sub-columns of a ***Conformance Table***.
Can Write: succinctly specifies whether the ***Producer*** is capable of writing a specified ***Trait***.
Does Write: succinctly specifies whether the ***Producer*** actually writes a specified ***Trait***.

Table 26: Conformance Requirements for Writing

Value	Can Write	Does Write	Description
`w`	SHALL	SHALL	The ***Producer*** SHALL write the ***Trait***. If an Attribute has a specified default value (as defined in [JDF1.5]) the ***Producer*** SHALL supply the Attribute when its value is the default value. If the EBNF in the *Name* column of [JDF1.5] specifies "*" or "+", a "`w`" means that the ***Producer*** SHALL write at least one ***Trait***. **Note:** an ***ICS*** uses the "`w`" for an ***Attribute Value*** only if it is the sole value allowed for a ***Conformance Level*** or if it is a required NMTOKEN for an NMTOKENS value or a required enumeration for an enumerations value.
`w←`	SHALL	MAY	The ***Producer*** SHALL write the ***Trait***, if some runtime condition is met. The condition is always specified in the Description column of the ***Conformance Table*** with the following exceptions: • If an ***Attribute Value*** has no explicit condition, the implicit condition is "as specified by [JDF1.5]", which means, "write one of the values". • If a ***Trait*** has no explicit condition, and [JDF1.5] provides no explicit condition, then the condition is "the ***Producer*** SHALL be able to be configured to write the ***Trait***." A ***Producer*** determines if some runtime condition is met by executing some software test. For example, the ***Producer***'s test might: • Request that a human operator select some option in a GUI. • Determine if some other JDF Attribute has a certain value. • Determine if some value is present in some file. • Query the network for the presence of some ***Device***. **Note:** all of these examples illustrate testable conditions. If an Attribute has a specified default value (as defined in [JDF1.5]) the ***Producer*** SHALL supply the Attribute when its value is the default value. **Note:** "B← A" in logic means "B if A". So "`w←`" means "write if some condition in the Description column is met".
`w?`	MAY	MAY	The ***Producer*** MAY write the ***Trait***.
`!w`	SHALL NOT	SHALL NOT	The ***Producer*** SHALL NOT create, modify or erase the ***Trait***. **Note:** "!A" in programming means "not A". So "`!w`" means "don't write".
			The ***Conformance Table*** cell is left blank with regard to writing but has an "`r`", "`r←`" or "`r?`". Such a cell has an implicit "`w?`" unless prohibited by [JDF1.5].

A.2.2.3 Conformance Requirements for Reading

Table 27 lists the ***Conformance Requirements*** for reading in descending order of strength from "`r`" (the strongest) to "`r?`" (the weakest).

Table 27: Conformance Requirements for Reading

Value	Description
r	The ***Consumer*** SHALL read and ***Support*** the ***Trait***. • If an Attribute is marked with an "r" and there are no values rows listed under it in the ***ICS***, then the ***Consumer*** SHALL ***Support*** at least one value. • All tables with ***Attribute Value*** rows and ***Attribute*** rows with no values listed where "r" values appear will specify the description of a test that can be applied to determine if the ***Trait*** is properly ***Supported***. These tests descriptions are preceded by "**r-Test:**". If the **r-Test** for an ***Attribute Value*** is the same as the **r-Test** specified for the ***Attribute*** row, then the ***Attribute Value*** row NEED NOT specify an **r-Test**.
r←	The ***Consumer*** SHALL read and ***Support*** the ***Trait***, if some testable condition is met. The condition is always specified in the Description column. **Note:** "B← A" in logic means "B if A". So "r←" means "read and ***Support*** if some condition in the Description column is met". All tables with ***Attribute Value*** rows and ***Attribute*** rows with no values listed where "r←" values appear will specify the description of a test that can be applied to determine if the ***Trait*** is properly ***Supported***. These tests descriptions are preceded by "**r-Test:**". If the **r-Test** for an ***Attribute Value*** is the same as the **r-Test** specified for the ***Attribute*** row, then the ***Attribute Value*** row NEED NOT specify an **r-Test**.
r?	The ***Consumer*** MAY read and ***Support*** the ***Trait***. The ***Trait*** MAY be ***Supported***, but if ***Supported***, any contained ***Traits*** SHALL be ***Supported*** according to their ***Conformance Requirements***.
	The ***Conformance Table*** cell is left blank with regard to reading but has a "w", "w←", "w?" or "!w". Such a cell has an implicit "r?".

Appendix B: Changes from Base ICS 1.4

This appendix lists the changes made to Base ICS 1.4 to make Base ICS 1.5:

Table 28: Changes from Base ICS 1.4

Location	Description
1. Table 7: JDF Node	Deleted @*Activation* values: "*TestRun*" and "***all remaining values***".
2. Table 8: ResourcePool	Added @*Class* and @*ID*. Required in [JDF1.5].
3. Table 16: Abstract Partitionable Resource	Added @*Class*.
4. Abstract ResoureLink	PhysicalLink renamed ResourceLink and table merged into Table 17: ResourceLink and ResourceLink/AmountPool/PartAmountResourceLink.
5. Table 23: NodeInfo	Deleted @*Route*.