


JDF Specification

Release 1.6

September 25, 2015

Draft 3

CIP4 THANKS ITS PARTNER LEVEL MEMBERS



Kodak



HEIDELBERG

MULLER MARTINI

RICOH

xerox



Legal Notice

Use of this document is subject to the following conditions which are deemed accepted by any person or entity making use hereof.

Copyright Notice

Copyright © 2000-2015, International Cooperation for the Integration of Processes in Prepress, Press and Postpress (CIP4) with registered office in Zurich, Switzerland. All Rights Reserved. CIP4 hereby grants to any person or entity obtaining a copy of the Specification and associated documentation files (the “Specification”) a perpetual, worldwide, non-exclusive, fully paid-up, royalty-free copyright license to use, copy, publish, distribute, publicly display, publicly perform, and/or sublicense the Specification in whole or in part verbatim and without modification, unless otherwise expressly permitted by CIP4, subject to the following conditions. This legal notice SHALL be included in all copies containing the whole or substantial portions of the Specification. Copies of excerpts of the Specification which do not exceed five (5) pages SHALL include the following short form Copyright Notice: Copyright © 2000-2013, International Cooperation for the Integration of Processes in Prepress, Press and Postpress (CIP4) with registered office in Zurich, Switzerland.

Trademarks and Tradenames

International Cooperation for the Integration of Processes in Prepress, Press and Postpress, CIP4, Job Definition Format, **XJDF**, Job Messaging Format, JMF, XJDF, XJMF and the CIP4 logo are trademarks of CIP4. Rather than put a trademark symbol in every occurrence of other trademarked names, we state that we are using the names only in an editorial fashion, and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Except as contained in this legal notice or as allowed by membership in CIP4, the name of CIP4 SHALL not be used in advertising or otherwise to promote the use or other dealings in this Specification without prior written authorization from CIP4.

Waiver of Liability

The XJDF Specification is provided as is, without warranty of any kind, express, implied, or otherwise, including but not limited to the warranties of merchantability, fitness for a particular purpose and noninfringement. In no event will CIP4 be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of, or in connection with the XJDF Specification or the use or other dealings in the XJDF Specification.

Table of Contents

Front Matter

Legal Notice	i
Table of Contents	iii
JDF Preface and User Overview	xlv
Chapter 1 Introduction	1
1.1 Background on XJDF	1
1.2 Design Criteria for XJDF	1
1.2.1 Simplify and reduce variations	2
1.2.1.1 Reduce the barrier of entry	2
1.2.1.2 Replace abstract data types with explicit elements and children	2
1.2.1.3 Remove ResourceLinks	2
1.2.2 Enable dynamic changes	2
1.2.2.1 Remove schema defaults.	2
1.2.2.3 Retain the semantic structures	2
1.2.2.4 Remove implementation specific details.....	2
1.2.4.1 Spawning and Merging.....	3
1.2.2.5 Enhance Compatibility with standard XML and XML Tools	3
1.2.2.6 Partitioning and Inheritance	3
1.2.6.1 Removal of Partition SignatureName	3
1.2.2.7 Device Capabilities	3
1.3 Background on JDF	3
1.4 Document References	4
1.5 Conventions Used in This Specification	4
1.5.1 Text Styles	4
1.5.2 XPath Notation Used in this Specification.....	5
1.5.3 Modification Notes	6
1.5.3.1 Location of Modification Notes.....	6
1.5.4 Specification of Cardinality.....	7
1.5.5 Template for Narrative Description of Resources	7
1.5.6 Template for Tables that Describe Elements	8
1.6 Glossary	9
1.6.1 Conformance Terminology.....	16
1.6.2 Conformance Requirements for JDF XJDF Entities	17
1.6.2.1 Conformance Requirements for Support of Attributes and Attribute Values.....	17
1.6.2.2 Conformance Requirements for Support of Elements	18

Table of Contents

1.6.2.3 Conformance Requirements for Support of Processes	18
1.6.2.4 Conformance Requirements for Support of Combined Processes.....	18
1.6.3 Conformance to Settings Policy.....	19
1.6.4 Interoperability Conformance Specifications.....	19
1.7 Data Structures	19
1.8 Units	22
1.8.1 Counting in JDF XJDF	23
Chapter 2 Overview of JDF XJDF	25
2.1 System Components	25
2.1.1 Job Components.....	25
2.1.1.1 Jobs and Nodes	25
2.1.1.2 Elements.....	25
2.1.1.3 Attributes	25
2.1.1.4 Relationships.....	25
2.1.1.5 Links	26
2.1.2 Referencing External Data.....	26
2.1.3 Identifying Sections of XJDF from External Sources	26
2.1.4 Identifying Sections of XJDF from within the Same XJDF	26
2.1.5 Workflow Component Roles	26
2.1.5.1 Machines	27
2.1.5.2 Devices.....	27
2.1.5.3 Agents	27
2.1.5.4 Queue	27
2.1.5.5 Controllers	28
2.1.5.6 Management Information Systems—MIS	28
2.1.5.7 System Interaction	28
2.2 JDF XJDF Workflow	30
2.2.1 Product Intent and Processes.....	31
2.2.2 Job Structure.....	31
2.3 Hierarchical Tree Structure and Networks in JDF	33
2.4 Role of Messaging in JDF XJDF	35
2.5 Coordinate Systems in JDF XJDF	35
2.5.1 Introduction	35
2.5.1.1 Source Coordinate Systems	37
2.5.2 Coordinates and Transformations.....	37
2.5.3 Coordinate Systems of Resources and Processes.....	38
2.5.3.1 Coordinate Systems of Combined Processes.....	38
2.5.3.2 Coordinate System Transformations	38
2.5.4 Product Example: Simple Brochure	40
2.5.5 General Rules	46

2.5.6 Homogeneous Coordinates	46
Chapter 3 Structure of XJDF JDF Nodes and Jobs	49
3.1 Modification Notes	50
3.2 Generic Contents of All Elements	50
3.2.1 Structure Diagram.....	52
3.3 JDF XJDF Node	53
3.3.1 Modification Notes	53
3.3.2 XJDF Node	53
3.3.3 Structure Diagram of XJDF JDF Node.....	61
3.4 Common Elements	63
3.4.1 Comment	63
3.4.2 GeneralID.....	66
3.5 Common Node Types	67
3.5.1 Product Intent Nodes	68
3.5.1.1 Process Group Nodes.....	69
3.5.1.2 Use of the Types Attribute in Process Group Nodes – Gray Boxes	69
3.5.2 Process Nodes.....	70
3.5.2.1 Use of NamedFeature in Product and Process Group Nodes	70
3.5.2.2 Specifying NamedFeatures with GeneralID	70
3.5.2.3 ResourceLink Structure in Process Group Nodes.....	70
3.5.3 Combined Process Nodes	72
3.5.3.1 Combined Process Nodes with Multiple Processes of the Same Type.....	72
3.5.3.2 Specifying non-linear dependencies in a Combined Process Node.....	73
3.5.4 Process Nodes.....	75
3.6 AncestorPool	75
3.6.1 Ancestor.....	75
3.7 CustomerInfo	77
3.8 NodeInfo	77
3.9 StatusPool	77
3.10 ResourcePool and its Resource Children	77
3.10.1 ResourcePool	77
3.10.2 Resource.....	77
3.10.3 Abstract Resource	78
3.10.3.1 SourceResource	82
3.10.4 Structure Diagram.....	82
3.11 ProductList	83
3.12 ResourceSet	84
3.12.1 Structure Diagram of ResourceSet.....	85

3.12.2 Dependent	86
3.13 Subelements of Resource	87
3.13.1 Location	87
3.13.2 Part	88
3.13.2.1 Partitioned Subsets of ResourceSets	94
3.13.2.2 Selecting a Partition	94
3.13.2.3 Referencing Multiple Resources of the Same Type	94
3.13.2.4 Multiple Part Elements in One Resource	94
3.13.3 AmountPool	95
3.13.3.1 Structure Diagram of AmountPool	95
3.13.3.2 PartAmount	95
3.13.4 Resource Classes	97
3.13.4.1 Parameter Resource	97
3.13.4.1.1 Abstract Parameter Resource	98
3.13.4.2 Intent Resource	98
3.13.4.3 ImplementationResource	98
3.13.4.4 Consumable Resource	98
3.13.4.5 Quantity Resource	98
3.13.4.6 Handling Resource	98
3.13.4.7 PhysicalResource	99
3.13.4.7.1 Abstract PhysicalResource	99
3.13.4.7.2 Location	100
3.13.4.8 PlaceHolder Resource	101
3.13.5 Position of Resources within JDF Nodes	101
3.13.6 Pipe Resources	101
3.13.7 ResourceUpdate	101
3.14 ResourceLinkPool and ResourceLink	102
3.14.1 ResourceLinkPool	102
3.14.2 ResourceLink	102
3.14.3 Structure Diagram	105
3.14.3.1 AmountPool and PartAmount	112
3.14.3.1.1 AmountPool	112
3.14.3.1.2 PartAmount	112
3.14.4 Resource Amount	116
3.14.4.1 Specifying Amount for a Partially-Completed Process	116
3.14.5 Identification of PhysicalResources	116
3.14.5.0.1 Lot	117
3.15 ResourcePool and ResourceLinkPool – Deep Structure	118
3.15.1 ResourceElement – Subelement of a Resource	118
3.15.1.1 ResourceElement	118
3.15.1.2 Abstract ResourceElement	118
3.15.2 ResourceRef – Element for Inter-Resource Linking and refelement	118
3.15.2.1 ResourceRef	118

3.15.2.2 Abstract ResourceRef	118
3.15.2.3 ResourceRef Elements in the AncestorPool/Ancestor Element	119
3.15.2.4 Status of a Resource that Contains an rRef Reference	120
3.15.2.5 Alignment of ResourceLink and ResourceRef	120
3.15.3 Set of Resources and Partitioned Subsets Thereof.....	121
3.15.4 Resource Amount	121
3.15.4.1 Evaluating and Updating Amount-Related Attributes in a Device.....	122
3.15.4.2 Specifying Amount for a Partially-Completed Process	123
3.15.5 Description of Partitioned Resources.....	125
3.15.5.1 Subelements in Partitioned Resources.....	125
3.15.5.2 Amount in Partitioned Resources	126
3.15.5.3 Relating PartIDKeys and Partitions	127
3.15.5.3.1 Incomplete Partitions	127
3.15.5.3.2 Number of Partition Keys per Partitioned Leaf or Node	127
3.15.5.3.3 Degenerate Partitions	128
3.15.5.4 Partitioning of Resource Subelements	128
3.15.5.5 Logical Partitions and the Identical Element.....	130
3.15.5.5.1 Identical	130
3.15.5.5.2 Restrictions when using Identical Elements	131
3.15.6 PartIDKeys Attribute and Partition Keys	132
3.15.6.1 Partitionable Resource	132
3.15.6.2 Part	134
3.15.6.3 Options in Intent Resources	144
3.15.6.4 Locations of PhysicalResources	144
3.15.7 Linking to Resources	145
3.15.7.1 Linking to Subsets of Resources.....	146
3.15.7.2 Reordering the Processing of Resources	146
3.15.7.3 Handling Amount in a ResourceLink to a Partitioned Resource	146
3.15.7.4 Implicit, Sparse and Explicit PartUsage in Partitioned Resources	147
3.15.7.5 Referencing Multiple Resources of the Same Type	148
3.15.8 Splitting and Combining Resources.....	150
3.16 AuditPool and Audit	151
3.16.1 AuditPool.....	152
3.16.2 Structure Diagram.....	153
3.16.3 Abstract Audit.....	154
3.16.4 Audit.....	154
3.16.4.1 Structure Diagram of Audit Elements	155
3.16.4.2 Created	158
3.16.4.3 Deleted	158
3.16.4.4 Merged	158
3.16.4.5 Modified.....	159
3.16.4.6 AuditNotification	159
3.16.4.7 Notification	159
3.16.4.8 PhaseTime	163

Table of Contents

3.16.4.9 AuditStatus.....	163
3.16.4.9.1 Activity	166
3.16.4.9.2 ModulePhase	166
3.16.4.10 ProcessRun.....	168
3.16.4.11 AuditResource	169
3.16.4.12 ResourceAudit	169
3.16.4.12.1 Logging Machine Data by Using the ResourceAudit	172
3.16.4.12.2 Logging Changes in Product Descriptions by Using the ResourceAudit	173
3.16.4.13 Spawed.....	174
3.17 XJDF Extensibility	175
3.18 JDF Extensibility	175
3.18.1 Namespaces in XML	175
3.18.1.1 JDF XJDF Namespace.....	175
3.18.1.2 JDF Extension Namespace	175
3.18.2 Foreign Namespaces.....	175
3.18.3 Creating Extension Assets:.....	176
3.18.4 Extending Process Types	176
3.18.4.1 Rules about Process Extension	176
3.18.5 Extending the NodeInfo and CustomerInfo Nodes	177
3.18.6 Extending Existing Resources	177
3.18.7 Extending NMTOKEN Lists.....	177
3.18.8 Creating New Resources.....	177
3.18.9 Future JDF XJDF Extensions	178
3.18.10 Maintaining Extensions	178
3.18.11 Processing Unknown Extensions.....	178
3.18.12 Derivation of Types in XML Schema.....	178
3.19 JDF Versioning	178
3.19.1 JDF Versioning Requirements	178
3.19.2 JDF Version Definition	179
3.19.3 JDF Version Policies.....	179
3.19.3.1 JDF Specification Version Policies	179
3.19.3.2 JDF Schema Version Policies	180
3.19.3.3 JDF Application Version Policies.....	180
3.19.3.3.1 JDF Agent Version Policies	180
3.19.3.3.2 JDF Device/Controller Version Policies	180
Chapter 4 Life Cycle of JDF XJDF	183
4.1 Introduction	183
4.2 Creation and Modification	183
4.2.1 Product Intent Constructs	183
4.2.1.1 Representation of Product Intent	184
4.2.1.2 Representation of Product Binding.....	184

4.2.2 Specification of Delivery of End Products	184
4.2.3 Specification of Process Specifics for Product Intent Nodes	184
4.3 Process Routing	186
4.3.1 Determining Executable Nodes	186
4.3.2 Distributing Processing to Work Centers or Devices	187
4.3.3 Device / Controller Selection.....	188
4.4 Execution Model	188
4.4.1 Serial Processing	188
4.4.2 Partial Processing of Nodes with Partitioned Resources.....	190
4.4.3 Overlapping Processing Using Pipes	191
4.4.3.1 Dynamic Pipes	194
4.4.3.2 Pipes of Partitionable Resources.....	195
4.4.3.3 Example JMFPush Sequence.....	195
4.4.3.4 Comparison of Non-Dynamic and Dynamic Pipes.....	196
4.4.4 Parallel Processing	196
4.4.5 Iterative Processing	197
4.4.5.1 Informal Iterative Processing.....	197
4.4.5.2 Formal Iterative Processing	197
4.4.6 Approval, Quality Control and Verification	197
4.5 Spawning and Merging	198
4.5.1 Case 1: Standard Spawning and Merging	199
4.5.2 Case 2: Spawning and Merging with Resource Copying.....	201
4.5.2.1 Spawning of Resources with Inter-Resource Links.....	201
4.5.3 Case 3: Parallel Spawning and Merging of Partitioned Resources	202
4.5.4 Case 4: Nested Spawning and Merging in Reverse Sequence	202
4.5.5 Case 5: Spawning and Merging of Independent Jobs	203
4.5.6 Case 6: Simultaneous Spawning and Merging of Multiple Nodes	203
4.6 Node and Resource IDs	203
4.7 Error Handling	204
4.7.1 Classification of Notifications	204
4.7.2 Event Description.....	204
4.7.3 Error Logging in the JDF File	205
4.7.4 Error Handling via Messaging (JMF)	205
4.8 Test Running	205
4.8.1 Resource Status During a Test Run	205
4.9 Capability and Constraint Definitions	206
Chapter 5 JMF XJMF Messaging with the Job Messaging Format .	207
5.1 JMF XJMF Root	208

5.1.1 Message	210
5.1.2 Message	210
5.1.3 Structure Diagram.....	211
5.2 List of All JMF XJMF Messages	213
5.3 JMF XJMF Message Families	216
5.3.1 Query	216
5.3.1.1 Subscription	218
5.3.2 Command	220
5.3.3 Signal	222
5.3.3.1 Trigger	223
5.3.3.2 ChangedPath	224
5.3.4 Response.....	224
5.3.5 Acknowledge.....	226
5.3.6 Registration.....	229
5.4 JMF Handshaking	229
5.4.1 Single Query/Command Response Communication	230
5.4.2 Signal and Acknowledge Handshaking.....	230
5.4.3 Reliable Signalling	230
5.4.4 Persistent Channels.....	230
5.4.4.1 Persistent Channels for Signals.....	231
5.4.4.2 Persistent Channels for Commands	231
5.4.5 Subscription	231
5.4.5.1 ObservationTarget	232
5.4.6 Scope of Subscriptions	233
5.4.7 Deleting Persistent Channels.....	233
5.5 JMF Messaging Levels	233
5.6 Error and Event Messages	234
5.7 Message Template	234
5.7.1 Object Type Column	235
5.7.1.1 QueryTypeObj	235
5.7.1.2 CommandTypeObj.....	235
5.7.1.3 ResponseTypeObj.....	235
5.8 Messages for Events and Capabilities	236
5.8.1 Events	237
5.8.2 KnownControllers.....	237
5.8.3 KnownDevices	237
5.8.3.1 QueryKnownDevices.....	237
5.8.3.1.1 DeviceFilter	238
5.8.3.2 DeviceFilter	238
5.8.3.3 ResponseKnownDevices	239

5.8.3.3.1 DeviceList	239
5.8.3.4 DeviceList.....	239
5.8.3.5 SignalKnownDevices.....	240
5.8.4 KnownJDFServices.....	240
5.8.5 KnownMessages.....	240
5.8.5.1 QueryKnownMessages	240
5.8.5.1.1 KnownMsgQuParams	241
5.8.5.2 KnownMsgQuParams	241
5.8.5.3 ResponseKnownMessages.....	241
5.8.5.3.1 MessageService	242
5.8.5.4 MessageService	242
5.8.5.5 SignalKnownMessages.....	244
5.8.6 KnownSubscriptions	244
5.8.6.1 QueryKnownSubscriptions	245
5.8.6.1.1 SubscriptionFilter	245
5.8.6.2 SubscriptionFilter.....	245
5.8.6.3 ResponseKnownSubscriptions.....	246
5.8.6.3.1 SubscriptionInfo	246
5.8.6.4 SubscriptionInfo.....	247
5.8.6.5 SignalKnownSubscriptions.....	247
5.8.7 Notification	248
5.8.7.1 QueryNotification	248
5.8.7.1.1 NotificationFilter	249
5.8.7.2 NotificationFilter	249
5.8.7.3 ResponseNotification.....	250
5.8.7.4 SignalNotification.....	250
5.8.7.4.1 Notification	251
5.8.7.4.2 Error	254
5.8.7.4.3 Event	254
5.8.7.4.4 Milestone	254
5.8.8 RepeatMessages	255
5.8.9 RequestForAuthentication	255
5.8.9.1 RequestForAuthentication Command.....	255
5.8.9.1.1 AuthenticationCmdParams	256
5.8.9.1.2 Certificate	257
5.8.9.1.3 AuthenticationResp	257
5.8.9.2 RequestForAuthentication Query	257
5.8.9.2.1 AuthenticationQuParams	258
5.8.10 StopPersistentChannel	259
5.8.10.1 CommandStopPersistentChannel.....	259
5.8.10.1.1 StopPersChParams	260
5.8.10.2 StopPersChParams.....	260
5.8.10.3 ResponseStopPersistentChannel	260
5.9 Messages to Query/Command a Job, Device or Controller Relating to Jobs and Devices	261

Table of Contents

5.9.1 FlushResources	262
5.9.1.1 FlushResources Command	262
5.9.1.2 CommandFlushResources	262
5.9.1.3 FlushResources Query	263
5.9.1.4 QueryFlushResources	263
5.9.1.4.1 FlushResourceParams	264
5.9.1.5 ResponseFlushResources.....	264
5.9.1.5.1 FlushedResources	265
5.9.1.5.2 SignalFlushResources.....	265
5.9.2 ModifyNode.....	265
5.9.2.1 ModifyNode Command	265
5.9.2.2 ModifyNode Signal.....	265
5.9.2.2.1 ModifyNodeCmdParams	265
5.9.2.2.2 NewComment	266
5.9.3 NewJDF	267
5.9.3.1 NewJDF Query	267
5.9.3.1.1 NewJDFQuParams	267
5.9.3.2 NewJDF Command.....	267
5.9.3.2.1 NewJDFCmdParams	268
5.9.3.2.2 IDInfo	268
5.9.4 NodeInfo	269
5.9.5 Occupation.....	269
5.9.6 Resource.....	269
5.9.6.1 Resource Query.....	269
5.9.6.2 QueryResource.....	269
5.9.6.2.1 ResourceQuParams	270
5.9.6.3 Resource Command.....	274
5.9.6.4 CommandResource	274
5.9.6.4.1 ResourceCmdParams	276
5.9.6.5 ResponseResource	279
5.9.6.5.1 ResourceInfo	279
5.9.6.6 SignalResource	284
5.9.7 ResourcePull.....	284
5.9.7.1 ResourcePullParams	285
5.9.8 ShutDown	286
5.9.8.1 CommandShutDown.....	287
5.9.8.1.1 ShutDownCmdParams	287
5.9.8.2 ShutDownCmdParams.....	287
5.9.8.3 ResponseShutDown	288
5.9.9 Status.....	288
5.9.9.1 QueryStatus.....	288
5.9.9.1.1 StatusQuParams	289
5.9.9.2 StatusQuParams	289
5.9.9.3 ResponseStatus	291
5.9.9.3.1 DeviceInfo	291
5.9.9.4 DeviceInfo	291

5.9.9.4.1 Activity	294
5.9.9.5 JobPhase.....	294
5.9.9.6 ModuleStatus	297
5.9.9.7 SignalStatus	298
5.9.10 Track	299
5.9.11 UpdateJDF	299
5.9.11.1 UpdateJDF Command.....	299
5.9.11.2 UpdateJDF Signal	300
5.9.11.2.1 UpdateJDFCmdParams	300
5.9.11.2.2 CreateLink	301
5.9.11.2.3 CreateResource	301
5.9.11.2.4 MoveResource	301
5.9.11.2.5 RemoveLink	301
5.9.12 WakeUp	302
5.9.12.1 CommandWakeUp.....	303
5.9.12.1.1 WakeUpCmdParams	303
5.9.12.2 WakeUpCmdParams.....	303
5.9.12.3 ResponseWakeUp	303
5.10 Messages for Pipe Control	304
5.10.1 PipeControl	304
5.10.1.1 CommandPipeControl	305
5.10.2 PipeParams	305
5.10.3 Common PipeControl Element	305
5.10.3.1 PipeParams.....	305
5.10.3.2 @Operation = Close	307
5.10.4 PipeClose.....	307
5.10.4.1 @Operation = Pull	307
5.10.5 PipePull.....	307
5.10.5.1 @Operation = Push	308
5.10.6 PipePush.....	308
5.10.6.1 @Operation = Pause	309
5.10.7 PipePause.....	309
5.10.7.1 ResponsePipeControl.....	310
5.11 Queue Support	310
5.11.1 Queue Entry ID Generation	310
5.12 Messages for Queue Entry Handling	311
5.12.1 AbortQueueEntry	317
5.12.1.1 AbortQueueEntryParams	317
5.12.2 HoldQueueEntry	318
5.12.2.1 HoldQueueEntryParams	318
5.12.3 ModifyQueueEntry	318
5.12.3.1 CommandModifyQueueEntry	319
5.12.3.1.1 ModifyQueueEntryParams	319

5.12.3.2 ResponseModifyQueueEntry1	321
5.12.4 RemoveQueueEntry	321
5.12.4.1 RemoveQueueEntryParams	322
5.12.5 RequestQueueEntry	322
5.12.5.1 CommandRequestQueueEntry	322
5.12.5.1.1 RequestQueueEntryParams	323
5.12.5.2 RequestQueueEntryParams	323
5.12.5.3 ResponseRequestQueueEntry1	323
5.12.6 ResubmitQueueEntry	324
5.12.6.1 CommandResubmitQueueEntry	324
5.12.6.1.1 ResubmissionParams	325
5.12.6.2 ResubmissionParams	325
5.12.6.2.1 Referencing values for incremental update	325
5.12.6.2.1.1 Finding the correct XJDF node to update	325
5.12.6.2.1.2 Finding the correct Resource to update	325
5.12.6.2.2 Updating values	325
5.12.6.2.3 Removing values	325
5.12.6.3 ResponseResubmitQueueEntry1	326
5.12.7 ResumeQueueEntry	326
5.12.7.1 ResumeQueueEntryParams	327
5.12.8 ReturnQueueEntry	327
5.12.8.1 CommandReturnQueueEntry	328
5.12.8.1.1 ReturnQueueEntryParams	328
5.12.8.2 ReturnQueueEntryParams	328
5.12.8.3 ResponseReturnQueueEntry1	329
5.12.9 SetQueueEntryPosition	329
5.12.9.1 QueueEntryPosParams	329
5.12.10 SetQueueEntryPriority	330
5.12.10.1 QueueEntryPriParams	330
5.12.11 SubmitQueueEntry	330
5.12.11.1 CommandSubmitQueueEntry	331
5.12.11.1.1 QueueSubmissionParams	331
5.12.11.2 QueueSubmissionParams	331
5.12.11.3 ResponseSubmitQueueEntry	334
5.12.12 SuspendQueueEntry	335
5.12.12.1 SuspendQueueEntryParams	335
5.13 Messages for Global Handling of Queues	335
5.13.1 CloseQueue	337
5.13.2 FlushQueue	337
5.13.2.1 FlushQueue Command	337
5.13.2.1.1 FlushQueueParams	338
5.13.2.2 FlushQueue Query	338
5.13.2.2.1 FlushQueueInfo	339
5.13.3 HoldQueue	339
5.13.4 OpenQueue	339

5.13.5 QueueEntryStatus.....	339
5.13.6 QueueStatus	339
5.13.6.1 QueryQueueStatus	340
5.13.6.1.1 QueueStatusParams	340
5.13.6.2 ResponseQueueStatus	341
5.13.6.3 SignalQueueStatus	341
5.13.7 ResumeQueue.....	342
5.13.8 SubmissionMethods.....	342
5.13.8.1 SubmissionMethods.....	342
5.14 Elements for Queues	343
5.14.1 Queue	343
5.14.2 QueueEntry.....	345
5.14.3 QueueEntryDef	347
5.14.4 QueueFilter	348
5.15 Gang Jobs	349
5.15.1 ForceGang.....	350
5.15.1.1 CommandForceGang	350
5.15.1.1.1 GangCmdFilter	351
5.15.1.2 GangCmdFilter	351
5.15.1.3 ResponseForceGang	351
5.15.2 GangStatus	351
5.15.2.1 CommandGangStatus	351
5.15.2.1.1 GangQuFilter	352
5.15.2.2 GangQuFilter	352
5.15.2.3 ResponseGangStatus	352
5.15.2.3.1 GangInfo	352
5.15.2.4 GangInfo	352
5.16 Extending Messages	353
5.16.1 IfraTrack Support	354
Chapter 6 Processes	355
6.1 Combining Individual Process Steps	355
6.1.1 Nodes with Multiple Processes of the Same Type.....	355
6.2 Process Template	355
6.3 General Processes	358
6.3.1 Approval.....	358
6.3.2 Buffer	358
6.3.3 Combine.....	359
6.3.4 Delivery	359
6.3.5 ManualLabor	360
6.3.6 Ordering	360

6.3.7 Packing	360
6.3.8 QualityControl	360
6.3.9 ResourceDefinition.....	361
6.3.10 Split	361
6.3.11 Verification	361
6.4 Prepress Processes	363
6.4.1 AssetListCreation.....	363
6.4.2 Bending.....	363
6.4.3 ColorCorrection.....	364
6.4.4 ColorSpaceConversion	364
6.4.5 ContactCopying	365
6.4.6 ContoneCalibration	366
6.4.7 CylinderLayoutPreparation	366
6.4.8 DBDocTemplateLayout.....	367
6.4.9 DBTemplateMerging	367
6.4.10 DieDesign	367
6.4.11 DieLayoutProduction.....	367
6.4.12 DigitalDelivery	370
6.4.13 FilmToPlateCopying.....	370
6.4.14 FormatConversion	371
6.4.15 ImageEnhancement.....	371
6.4.16 ImageReplacement	371
6.4.17 ImageSetting.....	371
6.4.18 Imposition.....	372
6.4.18.1 Glossary for Automated Imposition	373
6.4.18.2 Variables for Automated Imposition	377
6.4.18.3 Execution Model for Automated Imposition	379
6.4.18.4 Configuration for Various Automated Impositions	382
6.4.18.4.1 Using Logical Stacks	382
6.4.18.4.1.1 Imposition for Cut and Stack	382
6.4.18.4.2 Imposition for Signatures with Saddle Stitching	383
6.4.18.4.3 Selecting from Multiple Imposition Templates When Processing an unstructured PDL	384
6.4.18.4.4 Imposition for Start of a Chapter	384
6.4.18.4.5 Imposition for Regenerating Sheet Surfaces	385
6.4.18.4.6 Imposition for Document-Major Processing of a VDP Structured PDL	385
6.4.19 InkZoneCalculation	385
6.4.20 Interpreting	386
6.4.20.1 RIPing	386
6.4.21 LayoutElementProduction	387
6.4.22 LayoutPreparation	387
6.4.23 LayoutShifting	388
6.4.24 PageAssigning	388

6.4.25 PDFToPSConversion.....	388
6.4.26 PDLCreation	389
6.4.27 Preflight.....	389
6.4.28 PreviewGeneration	390
6.4.29 Proofing.....	392
6.4.30 PSToPDFConversion.....	392
6.4.31 RasterReading	393
6.4.32 Rendering	393
6.4.33 RIPing	394
6.4.34 Scanning	395
6.4.35 Screening.....	395
6.4.36 Separation.....	396
6.4.37 SheetOptimizing.....	396
6.4.38 SoftProofing	397
6.4.39 Stripping.....	397
6.4.39.1 Pagination in Stripping	397
6.4.40 Tiling	399
6.4.41 Trapping.....	400
6.5 Press Processes	400
6.5.1 ConventionalPrinting.....	401
6.5.2 DigitalPrinting.....	403
6.5.3 Varnishing	405
6.5.4 IDPrinting	406
6.6 Postpress Processes	406
6.6.1 AdhesiveBinding	406
6.6.2 BlockPreparation.....	406
6.6.3 BoxFolding	406
6.6.4 BoxPacking	407
6.6.5 Bundling	408
6.6.6 CaseMaking	408
6.6.7 CasingIn	409
6.6.8 ChannelBinding	409
6.6.9 CoilBinding	410
6.6.10 Collecting	411
6.6.11 CoverApplication	411
6.6.12 Creasing.....	412
6.6.13 Cutting.....	412
6.6.14 DieMaking	413
6.6.15 Dividing	413
6.6.16 Embossing	414

Table of Contents

6.6.17 EndSheetGluing.....	414
6.6.18 Feeding	415
6.6.19 Folding	416
6.6.20 Gathering	416
6.6.21 Gluing.....	417
6.6.22 HeadBandApplication	418
6.6.23 HoleMaking	418
6.6.24 Inserting	418
6.6.25 Jacketing.....	419
6.6.26 Labeling	419
6.6.27 Laminating	420
6.6.28 LongitudinalRibbonOperations.....	420
6.6.29 LooseBinding	420
6.6.30 Binding Methods	421
6.6.30.1 Single-Leaf Binding Methods.....	421
6.6.30.2 Loose-Leaf Binding Method.....	421
6.6.30.3 Mechanical Binding Methods.....	421
6.6.31 Numbering	422
6.6.32 Palletizing.....	422
6.6.33 Perforating	423
6.6.34 PlasticCombBinding.....	423
6.6.35 PrintRolling.....	424
6.6.36 RingBinding.....	424
6.6.37 SaddleStitching	425
6.6.38 ShapeCutting	425
6.6.39 ShapeDefProduction.....	426
6.6.40 Shrinking	426
6.6.41 SideSewing	427
6.6.42 SpinePreparation	427
6.6.43 SpineTaping.....	427
6.6.44 Stacking	428
6.6.45 StaticBlocking	428
6.6.46 Stitching	428
6.6.47 Strapping.....	429
6.6.48 StripBinding.....	430
6.6.49 ThreadSealing.....	430
6.6.50 ThreadSewing.....	430
6.6.51 Trimming	431
6.6.52 WebInlineFinishing.....	431
6.6.53 Winding	432
6.6.54 WireCombBinding	432

6.6.55 Wrapping	433
6.7 Postpress Processes Structure	433
6.7.1 Block Production	433
6.7.1.1 Block Compiling	434
6.7.1.2 Block Joining	434
6.7.1.3 Binding Methods	434
6.7.1.3.1 Single-Leaf Binding Methods	434
6.7.1.3.2 Loose-Leaf Binding Method	434
6.7.1.3.3 Mechanical Binding Methods	434
6.7.2 HoleMaking	435
6.7.3 Laminating	435
6.7.4 Numbering	435
6.7.5 Packaging Processes	435
6.7.6 Processes in Hardcover Book Production	435
6.7.7 Sheet Processes	436
6.7.8 Tip-on/In	436
6.7.9 Trimming	436
6.7.10 Web Processes	436
Chapter 7 Product Intents	439
7.1 Product Description	439
7.1.1 Representation of Product Binding	439
7.1.2 Product	439
7.1.3 Intent	441
7.1.4 Product Intent	441
7.1.5 Structure Diagram of ProductList	442
7.1.6 Product Intent Descriptions	442
7.2 Intent Properties Template	443
7.2.1 Abstract Span Element	444
7.2.2 Span Elements	445
7.2.2.1 DurationSpan	446
7.2.2.2 EnumerationSpan	446
7.2.2.3 IntegerSpan	447
7.2.2.4 NameSpan	447
7.2.2.4.1 Specifying New Values in a NameSpan Subelement	447
7.2.2.5 NumberSpan	447
7.2.2.6 OptionSpan	448
7.2.2.7 ShapeSpan	448
7.2.2.8 StringSpan	449
7.2.2.9 TimeSpan	449
7.2.2.10 XYPairSpan	449
7.3 ArtDeliveryIntent	450

7.3.1 ArtDelivery	453
7.4 AssemblingIntent	456
7.4.1 AssemblyItem	457
7.4.2 BindIn	457
7.4.3 BlowIn	457
7.4.4 StickOn	458
7.5 BindingIntent	458
7.5.1 BindList	461
7.5.2 BindItem	462
7.5.3 AdhesiveBinding	463
7.5.4 BookCase	463
7.5.5 ChannelBinding	463
7.5.6 CoilBinding	463
7.5.7 EdgeGluing	463
7.5.8 HardCoverBinding	464
7.5.9 LooseBinding	467
7.5.10 PlasticCombBinding	468
7.5.11 RingBinding	468
7.5.12 SaddleStitching	469
7.5.13 SideSewing	470
7.5.14 SideStitching	470
7.5.15 SoftCoverBinding	470
7.5.16 StripBinding	472
7.5.17 Tabs	472
7.5.18 Tape	473
7.5.19 ThreadSealing	473
7.5.20 ThreadSewing	473
7.5.21 WireCombBinding	473
7.6 ColorIntent	474
7.6.1 SurfaceColor	478
7.6.2 ColorsUsed	481
7.7 ContentCheckIntent	481
7.7.1 ProofItem	482
7.8 DeliveryIntent	483
7.8.1 DropItemIntent	489
7.8.2 Pricing	489
7.8.3 Payment	490
7.8.4 CreditCard	490
7.9 EmbossingIntent	490

7.9.1 EmbossingItem	490
7.10 FoldingIntent	492
7.11 HoleMakingIntent	493
7.12 InsertingIntent	494
7.12.1 InsertList	495
7.12.2 Insert.....	495
7.13 LaminatingIntent	496
7.14 LayoutIntent	496
7.15 MedialIntent	500
7.16 NumberingIntent	508
7.17 PackingIntent	508
7.18 ProductionIntent	509
7.19 ProofingIntent	510
7.19.1 ProofItem	511
7.20 PublishingIntent	512
7.21 ScreeningIntent	513
7.22 ShapeCuttingIntent	514
7.22.1 ShapeCut.....	514
7.23 SizeIntent	515
7.24 VariableIntent	515
Chapter 8 Resources	517
8.1 Resource	517
8.1.1 Location	520
8.1.2 Specific Resourcer.....	520
8.2 AdhesiveBindingParams	520
8.3 ApprovalParams	520
8.3.1 ApprovalPerson	521
8.4 ApprovalSuccess	522
8.5 ApprovalDetails	522
8.5.1 ApprovalDetails.....	522
8.6 Assembly	523
8.6.1 AssemblySection	525
8.6.2 PageAssignedList	526
8.7 AssetListCreationParams	527

8.8 BendingParams	528
8.9 BinderySignature	528
8.9.1 Pagination for BinderySignatureType Fold	529
8.9.2 On the use of Bleed	534
8.9.3 On the use of Trim	534
8.9.4 SignatureCell	538
8.10 BlockPreparationParams	543
8.11 BoxFoldingParams	544
8.11.1 BoxApplication	546
8.11.2 BoxFoldAction	546
8.12 BoxPackingParams	550
8.13 BufferParams	551
8.14 Bundle	552
8.14.1 BundleItem	553
8.15 BundlingParams	555
8.16 ByteMap	555
8.16.1 Band	557
8.16.2 PixelColorant	557
8.17 CaseMakingParams	557
8.18 CasingInParams	559
8.19 ChannelBindingParams	560
8.20 CoilBindingParams	561
8.21 CollectingParams	562
8.22 Color	563
8.22.1 DeviceNColor	569
8.22.2 Diecutting Data (DDES3)	569
8.22.3 PrintConditionColor	570
8.23 ColorantControl	574
8.23.1 ColorantConvertProcess	580
8.23.2 ColorantOrder	580
8.23.3 ColorantParams	580
8.23.4 DeviceColorantOrder	580
8.23.5 ColorSpaceSubstitute	580
8.23.6 DeviceNSpace	582
8.24 ColorCorrectionParams	583
8.25 ColorPool	584

8.26 ColorSpaceConversionParams	585
8.27 Component	587
8.28 Contact	594
8.28.1 ComChannel.....	597
8.28.2 Company.....	598
8.28.3 Person.....	598
8.29 ContactCopyParams	599
8.30 ContentList	600
8.31 Content	600
8.31.1 ContentData.....	600
8.31.2 BarcodeProductionParams.....	603
8.31.3 ContentMetadata	604
8.31.4 PositionObj.....	604
8.32 ConventionalPrintingParams	608
8.33 CoverApplicationParams	611
8.33.1 Score.....	611
8.34 CreasingParams	613
8.35 CustomerInfo	614
8.35.1 CustomerMessage.....	615
8.36 CutBlock	615
8.37 CutMark	616
8.38 CuttingParams	618
8.38.1 CutBlock.....	618
8.39 CylinderLayout	619
8.39.1 CylinderPosition.....	620
8.40 CylinderLayoutPreparationParams	623
8.41 DBMergeParams	623
8.42 DBRules	623
8.43 DBSchema	624
8.44 DBSelection	624
8.45 DeliveryParams	624
8.45.1 Drop	628
8.45.2 DropItem	629
8.46 DevelopingParams	631
8.47 Device	631

8.47.1 IconList.....	635
8.47.2 Icon	635
8.47.3 Module	635
8.48 DieLayout	636
8.48.1 RuleLength.....	637
8.48.2 Station.....	638
8.49 DieLayoutProductionParams	638
8.49.1 RepeatDesc	639
8.50 DigitalDeliveryParams	643
8.51 DigitalMedia	644
8.52 DigitalPrintingParams	645
8.52.1 Coordinate systems in DigitalPrinting	645
8.53 DividingParams	650
8.54 ElementColorParams	650
8.55 EmbossingParams	652
8.55.1 Emboss	652
8.56 Employee	654
8.57 EndSheetGluingParams	654
8.57.1 EndSheet	655
8.58 ExposedMedia	656
8.59 ExternalImpositionTemplate	658
8.60 FeedingParams	658
8.60.1 Feeder.....	659
8.60.2 FeederQualityParams	660
8.60.3 CollatingItem.....	661
8.61 FileSpec	662
8.61.1 Container	669
8.61.2 Disposition	669
8.61.3 FileAlias	670
8.62 FoldingParams	671
8.63 FontParams	676
8.64 FontPolicy	676
8.65 FormatConversionParams	677
8.66 GatheringParams	677
8.67 GlueApplication	678

8.68 GluingParams	679
8.68.1 Glue	680
8.69 HeadBandApplicationParams	680
8.70 HoleList	681
8.71 HoleMakingParams	682
8.72 IdentificationField	684
8.72.1 BarcodeDetails.....	688
8.72.2 ExtraValues.....	689
8.72.3 Usage of barcode Attributes	689
8.73 IDPrintingParams	691
8.74 ImageCompressionParams	692
8.74.1 ImageCompression.....	692
8.74.2 CCITTFaxParams.....	695
8.74.3 DCTParams	695
8.74.4 FlateParams.....	697
8.74.5 JBIG2Params.....	697
8.74.6 JPEG2000Params	698
8.74.7 LZWParams	698
8.75 ImageEnhancementParams	699
8.75.1 ImageEnhancementOp	699
8.76 ImageReplacementParams	700
8.77 ImageSetterParams	702
8.78 Ink	704
8.79 InkZoneCalculationParams	706
8.80 InkZoneProfile	707
8.81 InsertingParams	707
8.82 InterpretedPDLData	710
8.83 InterpretingParams	710
8.83.1 InterpretingDetails.....	712
8.83.2 PDFInterpretingParams	713
8.83.3 OCGControl	714
8.83.4 ReferenceXObjParams	715
8.83.5 PDLResourceAlias	715
8.83.6 More about PDFInterpretingParams	716
8.83.6.1 PDF Optional Content Groups.....	716
8.84 JacketingParams	716

8.85 LabelingParams	718
8.86 LaminatingParams	718
8.87 Layout	719
8.87.1 ColorControlStrip	725
8.87.2 CIELABMeasuringField.....	726
8.87.3 DensityMeasuringField	728
8.87.4 DeviceMark	728
8.87.5 LayerList	732
8.87.6 LayerDetails	732
8.87.7 LogicalStackParams	733
8.87.8 Stack	733
8.87.9 PageCondition	734
8.87.10 SheetCondition	736
8.87.11 PlacedObject.....	737
8.87.12 PlacedObject.....	737
8.87.12.1 Abstract PlacedObject.....	737
8.87.13 ContentObject	741
8.87.14 MarkObject.....	747
8.87.15 CutMark	754
8.87.16 FillMark	755
8.87.17 MarkActivation	756
8.87.17.1 Dynamic Marks.....	756
8.87.18 DynamicField	757
8.87.19 More about Layout.....	758
8.87.19.1 Migrating from a Pre-JDF 1.3 Layout to a Partitioned Layout	758
8.87.19.1.1 Partition Key restrictions:	759
8.87.19.1.2 Position of PlacedObject Elements in Layout	759
8.87.19.2 CTM Definitions	760
8.87.19.3 Finding the Trim Box of an Object.....	760
8.87.19.4 Using Ord to Reference Elements in RunList Resources	761
8.87.19.5 Using Expressions in the OrdExpression Attribute	762
8.87.20 Signature.....	763
8.88 LayoutElement	763
8.88.1 Dependencies	767
8.89 LayoutElementProductionParams	768
8.89.1 LayoutElementPart	770
8.89.2 BarcodeProductionParams	771
8.89.3 PositionObj.....	771
8.90 LayoutPreparationParams	777
8.90.1 PageCell	787

8.90.2 ImageShift.....	789
8.91 LayoutShift	793
8.91.1 ShiftPoint.....	793
8.92 LongitudinalRibbonOperationParams	794
8.93 LooseBindingParams	794
8.93.1 ChannelBindingDetails.....	797
8.93.2 CoilBindingDetails.....	798
8.93.3 CombBindingDetails	798
8.93.4 RingBindingDetails.....	799
8.94 ManualLaborParams	800
8.95 Media	800
8.95.1 TabDimensions	814
8.95.2 More about Media	817
8.95.2.1 Inside Loss and Outside Gain	817
8.95.2.2 Corrugated Media:	818
8.95.2.3 Self adhesive Media	819
8.95.2.4 Flexo Plate Media	819
8.95.2.5 Flexo Sleeve Media	819
8.96 MediaSource	820
8.97 MiscConsumable	820
8.98 NodeInfo	821
8.99 NumberingParams	825
8.100 OrderingParams	825
8.101 PackingParams	825
8.102 PageAssignParams	825
8.103 PageList	826
8.103.1 PageData.....	827
8.103.2 PageElement	830
8.104 Pallet	831
8.105 PalletizingParams	832
8.106 PDFToPSConversionParams	833
8.107 PDLCreationParams	836
8.107.1 FontParams	837
8.107.2 PDFToPSConversionParams	837
8.107.3 PSCreationDetails.....	837
8.107.4 PSToPDFConversionParams	840
8.107.5 PDFCreationDetails	840

8.107.6 AdvancedParams.....	842
8.107.7 PDFXParams	844
8.107.8 ThinPDFParams	846
8.108 PDLResourceAlias	846
8.109 PerforatingParams	847
8.110 PlaceHolderResource	847
8.111 PlasticCombBindingParams	847
8.112 PlateCopyParams	848
8.113 PreflightAnalysis	848
8.114 PreflightInventory	849
8.115 PreflightParams	849
8.115.1 PreflightTest.....	849
8.115.2 PreflightAction.....	850
8.115.3 BasicPreflightTest.....	851
8.115.4 PreflightArgument.....	852
8.115.5 BoxArgument	852
8.115.6 BoxToBoxDifference	853
8.116 PreflightProfile	854
8.117 PreflightReport	854
8.117.1 PreflightCheck.....	855
8.117.2 PRItem.....	855
8.117.3 PRError.....	856
8.117.4 PRGroup.....	857
8.117.5 Abstract PRGroupOccurrenceBase	859
8.117.6 PRGroupOccurrenceBase	859
8.117.7 ArgumentValue	859
8.117.8 PRGroupOccurrence	859
8.117.9 StringListValue	860
8.117.10 PROccurrence	860
8.118 PreflightReportRulePool	860
8.118.1 PRRule.....	861
8.118.2 PRRuleAttr	861
8.119 Preview	862
8.120 PreviewGenerationParams	864
8.121 PrintCondition	866
8.122 PrintRollingParams	867

8.123 ProductionPath	868
8.123.1 FolderSuperstructureWebPath	869
8.123.2 PostPressComponentPath	869
8.123.3 PrintingUnitWebPath	869
8.124 ProofingParams	870
8.125 PSToPDFConversionParams	870
8.125.1 AdvancedParams	872
8.125.2 PDFXParams	874
8.125.3 ThinPDFParams	876
8.126 QualityControlParams	877
8.126.1 BindingQualityParams	877
8.127 QualityControlResult	877
8.127.1 QualityMeasurement	878
8.127.2 BindingQualityMeasurement	878
8.128 RasterReadingParams	879
8.129 RegisterMark	880
8.130 RenderingParams	881
8.130.1 TIFFFormatParams	882
8.130.2 TIFFtag	884
8.130.3 TIFFEmbeddedFile	884
8.131 ResourceDefinitionParams	884
8.131.1 ResourceParam	885
8.132 RingBindingParams	885
8.133 RollStand	887
8.134 RunList	887
8.134.1 Reordering the Processing of Resources	898
8.134.2 ByteMap	899
8.134.3 Band	900
8.134.4 DynamicInput	900
8.135 SaddleStitchingParams	903
8.136 ScanParams	903
8.137 ScavengerArea	904
8.138 ScreeningParams	905
8.139 SeparationControlParams	906
8.140 Shape	906
8.141 ShapeCuttingParams	907

8.142 ShapeDef	908
8.142.1 CutLines.....	910
8.143 ShapeDefProductionParams	910
8.143.1 ObjectModel.....	911
8.143.2 ShapeTemplate.....	911
8.143.3 ShapeDimension.....	912
8.144 Sheet	914
8.145 SheetOptimizingParams	914
8.145.1 GangElement.....	914
8.145.2 SeparationListBack	917
8.145.3 SeparationListFront.....	918
8.146 ShrinkingParams	918
8.147 SideSewingParams	918
8.148 SpinePreparationParams	918
8.149 SpineTapingParams	920
8.150 StackingParams	921
8.151 StaticBlockingParams	925
8.152 StitchingParams	925
8.153 Strap	929
8.154 StrappingParams	930
8.155 StripBindingParams	931
8.156 StrippingParams	932
8.156.1 ExternallImpositionTemplate	936
8.156.2 Position	937
8.156.3 StripCellParams	938
8.156.4 StripMark.....	941
8.157 Surface	946
8.158 ThreadSealingParams	946
8.159 ThreadSewingParams	946
8.160 Tile	948
8.161 Tool	949
8.162 TransferCurve	950
8.163 TransferCurvePool	951
8.163.1 TransferCurveSet.....	952
8.164 TransferFunctionControl	952

8.165 TrappingDetails	953
8.165.1 TrappingOrder.....	954
8.166 TrappingParams	954
8.166.1 ColorantZoneDetails	958
8.167 TrapRegion	958
8.168 TrimmingParams	959
8.169 UsageCounter	960
8.170 VarnishingParams	961
8.171 VerificationParams	962
8.172 WebInlineFinishingParams	963
8.172.1 FolderProduction.....	964
8.173 WindingParams	964
8.174 WireCombBindingParams	965
8.175 WrappingParams	966
Chapter 9 Device Capabilities	969
9.1 Device Capability Definitions	969
9.1.1 DeviceCap	969
9.1.1.1 Structure Diagram of DeviceCap.....	973
9.1.2 ActionPool.....	974
9.1.2.1 Action.....	975
9.1.3 DevCapPool.....	975
9.1.4 StatePool	975
9.1.5 ModulePool.....	976
9.1.5.1 ModuleCap.....	976
9.1.6 DevCaps	976
9.1.6.1 Loc	979
9.1.7 DevCap	980
9.1.8 State.....	981
9.1.8.1 Abstract State Element.....	983
9.1.8.2 State Elements.....	986
9.1.8.2.1 Structure Diagram of Specific State	986
9.1.8.2.2 BooleanState	988
9.1.8.2.2.1 ValueLoc	989
9.1.8.2.3 DateTimeState	989
9.1.8.2.4 DurationState	991
9.1.8.2.5 ElementState	993
9.1.8.2.6 EnumerationState	994
9.1.8.2.7 IntegerState	995
9.1.8.2.8 MatrixState	998
9.1.8.2.8.1 Value	1000

Table of Contents

9.1.8.2.9 NameState	1001
9.1.8.2.10 NumberState	1002
9.1.8.2.11 PDFPathState	1004
9.1.8.2.11.1 Value	1005
9.1.8.2.12 RectangleState	1005
9.1.8.2.13 ShapeState	1007
9.1.8.2.14 StringState	1009
9.1.8.2.14.1 Value	1011
9.1.8.2.15 XYPairState	1012
9.1.9 DisplayGroupPool	1014
9.1.9.1 DisplayGroup.....	1014
9.1.10 FeaturePool	1014
9.1.11 MacroPool.....	1015
9.1.11.1 macro	1016
9.1.11.2 choice	1016
9.1.11.3 otherwise.....	1017
9.1.11.4 when.....	1017
9.1.11.5 set	1017
9.1.11.6 FeatureAttribute	1017
9.1.11.7 call.....	1018
9.1.12 Performance	1018
9.1.13 TestPool.....	1019
9.1.13.1 Test.....	1019
9.1.14 Term.....	1019
9.1.14.1 Abstract Term	1019
9.1.14.2 Term Elements	1020
9.1.14.2.1 Structure Diagram of Term	1021
9.1.14.3 and.....	1022
9.1.14.4 or	1022
9.1.14.5 xor	1023
9.1.14.6 not	1023
9.1.14.7 TestRef.....	1023
9.1.14.8 Evaluation	1023
9.1.14.8.1 Abstract Evaluation	1024
9.1.14.8.2 Evaluation Elements	1025
9.1.14.8.2.1 Structure Diagram of Evaluation	1026
9.1.14.8.2.2 BooleanEvaluation	1027
9.1.14.8.2.3 DateTimeEvaluation	1028
9.1.14.8.2.4 DurationEvaluation	1028
9.1.14.8.2.5 EnumerationEvaluation.....	1028
9.1.14.8.2.6 IntegerEvaluation	1029
9.1.14.8.2.7 IsPresentEvaluation.....	1029
9.1.14.8.2.8 MatrixEvaluation	1029
9.1.14.8.2.8.1 Value	1030
9.1.14.8.2.9 NameEvaluation	1030
9.1.14.8.2.10 NumberEvaluation	1031
9.1.14.8.2.11 PDFPathEvaluation	1031
9.1.14.8.2.11.1 Value	1031

9.1.14.8.2.12 RectangleEvaluation	1031
9.1.14.8.2.13 ShapeEvaluation.....	1032
9.1.14.8.2.14 StringEvaluation.....	1032
9.1.14.8.2.14.1 Value	1033
9.1.14.8.2.15 XYPairEvaluation	1033
9.1.15 Examples of Device Capabilities.....	1033
9.1.15.1 Device Description of a Scanner.....	1033
9.1.15.2 Device Description of a Scanner #2.....	1037
9.2 Concept of the Preflight Process	1042
9.2.1 Object Classes.....	1042
9.2.1.1 Properties Implemented by each Class of Object	1043
9.2.1.2 Checking for the Presence of a Property.....	1044
9.2.1.3 Basic tests on set of objects	1045
9.2.2 Properties.....	1045
9.2.2.1 Annotation Properties	1046
9.2.2.2 Box Properties.....	1047
9.2.2.3 Class Properties.....	1047
9.2.2.4 Colorant Properties	1048
9.2.2.5 Document Properties.....	1049
9.2.2.6 Fill Properties.....	1053
9.2.2.7 Font Properties	1054
9.2.2.8 Graphic Properties	1055
9.2.2.9 Image Properties	1056
9.2.2.10 Logical Properties	1059
9.2.2.11 PageBox Properties	1059
9.2.2.12 Pages Properties	1059
9.2.2.13 PDLObject Properties	1060
9.2.2.14 Reference Properties	1061
9.2.2.15 Shading Properties	1061
9.2.2.16 Stroke Properties	1062
9.2.2.17 Text Properties	1062
9.2.2.18 Vector Properties	1063
Chapter 10 Subelements	1065
10.1 Address	1065
10.2 ApprovalPerson	1065
10.3 AutomatedOverPrintParams	1066
10.4 BarcodeCompParams	1067
10.5 BarcodeReproParams	1067
10.6 ColorantAlias	1068
10.7 ColorCorrectionOp	1069
10.8 ColorMeasurementConditions	1070

10.9 ColorSpaceConversionOp	1071
10.10 ComChannel	1079
10.11 ConvertingConfig	1081
10.12 CostCenter	1081
10.13 Crease	1082
10.14 Cut	1083
10.15 DeviceMark	1084
10.16 DeviceNSpace	1087
10.17 Disjointing	1088
10.18 Disposition	1090
10.19 FitPolicy	1090
10.20 Fold	1092
10.21 GlueLine	1093
10.22 HolePattern	1094
10.23 HoleLine	1096
10.24 InsertSheet	1097
10.25 JobField	1101
10.26 MarkColor	1102
10.27 MediaLayers	1102
10.28 MetadataMap	1102
10.28.1 Expr.....	1105
10.29 MISDetails	1108
10.30 ObjectResolution	1109
10.31 OCGControl	1110
10.32 Perforate	1110
10.33 Person	1111
10.34 RefAnchor	1112
10.35 RegisterRibbon	1113
10.36 ScreenSelector	1114
10.37 SeparationSpec	1116
Chapter 11 Building a System Around JDF XJDF	1119
11.1 Implementation Considerations and Guidelines	1119

11.2 Status Transitions	1119
11.3 Execution Model	1120
11.3.1 Serial Processing	1120
11.3.2 Partial Processing of Nodes with Partitioned Resources	1121
11.3.3 Parallel Processing	1121
11.3.4 Overlapping Processing	1121
11.3.4.1 Dynamic Pipes	1124
11.3.4.2 Example JMFPush Sequence	1125
11.3.4.3 Comparison of Non-Dynamic and Dynamic Pipes	1126
11.3.4.4 Metadata in Pipe Messages	1126
11.3.5 Approval, Proofing, Quality Control and Verification	1126
11.3.6 Error Handling	1126
11.3.7 Classification of Notifications	1126
11.3.8 Event Description	1127
11.3.9 Error Logging in the JDF File	1127
11.3.10 Error Handling via Messaging (XJMF)	1127
11.4 JDF XJDF and JMF XJMF Interchange Protocol	1127
11.4.1 File-Based Protocol (JDF)	1127
11.4.2 HTTP-Based Protocol (JDF + JMF)	1127
11.4.2.1 Protocol Implementation Details	1127
11.4.3 HTTP Port	1127
11.4.4 HTTP Request Method	1127
11.4.5 HTTPS-Based Protocol – SSL with two-way authentication	1128
11.4.5.1 Purpose	1128
11.4.5.2 Certificates	1128
11.4.5.2.1 Verification of Certificates	1129
11.4.5.2.3 Exchange of Certificates	1129
11.4.5.4 Standards	1131
11.4.5.5 Implementation	1131
11.4.5.5.1 Discovery Messages	1131
11.4.5.5.2 Example of Java Keytool Usage	1131
11.5 XJMF Handshaking	1131
11.5.1 Single Query/Command Response Communication	1131
11.5.1.1 XJMF Error Handling	1131
11.5.2 Subscribing for Signals	1132
11.5.3 Managing Persistent Channels	1133
11.5.4 Signal Handshaking	1133
11.5.5 Reliable Signalling	1133
11.5.5.1 12.3.4.1 Sequence of Signals	1133
11.5.6 Deleting Persistent Channels	1134
11.5.7 XJMF Bootstrapping	1134
11.5.8 Device / Controller Selection	1134

11.6 JDF XJDF Packaging	1134
11.6.1 MIME Basics	1135
11.6.2 MIME Types and File Extensions	1135
11.6.3 ZIP Packaging.....	1135
11.6.3.1 Identifying the Root XJMF.....	1136
11.6.3.2 Referencing Assets within a ZIP Package.....	1136
11.6.3.3 MIME Headers	1136
11.6.3.3.1 Content-Type Header	1136
11.6.3.3.2 Content-ID Header	1136
11.6.3.3.3 Content-Transfer-Encoding	1136
11.6.3.3.4 Content-Disposition Header	1137
11.6.3.4 CID URL Scheme.....	1137
11.6.3.5 Ordering of Body Parts in MIME Multipart/Related.....	1138
11.7 MIS Requirements	1139
11.8 Job Modification	1139
11.8.1 Rescheduling with ModifyQueueEntry	1140
11.8.2 Modifying Jobs	1140
11.9 Interoperability Conformance Specifications	1140
11.10 Use of XML Schema for Capability Descriptions	1141
Appendix A Encoding.....	1143
A.1 Notes About Encoding	1143
A.1.1 List, Range and Range List Data Types.....	1143
A.1.2 Whitespace.....	1143
A.1.3 Infinity Limits.....	1143
A.2 Simple Types — Attribute Values	1143
A.2.1 boolean.....	1143
A.2.2 CMYKColor.....	1143
A.2.3 date.....	1144
A.2.4 dateTime.....	1144
A.2.5 DateTimeRange	1144
A.2.6 DateTimeRangeList.....	1145
A.2.7 double.....	1145
A.2.8 DoubleList.....	1145
A.2.9 DoubleRange.....	1145
A.2.10 DoubleRangeList.....	1146
A.2.11 duration.....	1146
A.2.12 DurationRange	1146
A.2.13 DurationRangeList.....	1146
A.2.14 gYearMonth.....	1147

A.2.15 hexBinary.....	1147
A.2.16 ID	1147
A.2.17 IDREF.....	1147
A.2.18 IDREFS	1147
A.2.19 integer.....	1148
A.2.20 IntegerList.....	1148
A.2.21 IntegerRange.....	1148
A.2.22 IntegerRangeList	1148
A.2.23 LabColor.....	1149
A.2.24 language.....	1149
A.2.25 languages	1149
A.2.26 matrix.....	1149
A.2.27 NameRange	1150
A.2.28 NameRangeList.....	1150
A.2.29 NMTOKEN.....	1150
A.2.30 NMTOKENS	1150
A.2.31 PDFPath	1151
A.2.32 rectangle.....	1151
A.2.33 RectangleRange.....	1151
A.2.34 RectangleRangeList	1151
A.2.35 regExp	1152
A.2.36 shape.....	1152
A.2.37 ShapeRange.....	1152
A.2.38 ShapeRangeList	1152
A.2.39 sRGBColor	1153
A.2.40 string.....	1153
A.2.41 TimeRange	1153
A.2.42 TransferFunction	1153
A.2.43 URI	1154
A.2.44 URL	1154
A.2.45 XPath.....	1155
A.2.46 XYPair	1155
A.2.47 XYPairRange	1155
A.2.48 XYPairRangeList	1155
A.3 Enumerations and Lists	1156
A.3.1 enumeration.....	1156
A.3.2 enumerations.....	1156
A.3.3 Defined JDF XJDF enumeration Data Types	1156
A.3.3.1 Anchor.....	1156
A.3.3.2 JDFJMVersion	1156

A.3.3.3 NamedColor	1157
A.3.3.4 Orientation	1157
A.3.3.5 Sides	1158
A.3.3.6 Scope	1158
A.3.3.7 WorkStyle	1158
A.3.4 XYRelation.....	1159
A.4 JDF XJDF File Formats	1159
A.4.1 PNG Image Format	1159
Appendix B Schema	1161
B.1 Using xsi:type	1161
B.1.1 Using xsi:type with JDF XJDF Nodes.....	1161
B.1.2 Using xsi:type with JMF XJMF Messages	1162
Appendix C Supported String and NMOKEN values.....	1165
C.1 StatusDetails Supported Strings	1165
C.2 ModuleType Supported Strings	1170
C.3 NotificationDetails	1174
C.3.1 Abstract NotificationDetails	1174
C.3.2 NotificationDetails.....	1174
C.3.2.1 Barcode	1175
C.3.2.2 FCNKey	1175
C.3.2.3 SystemTimeSet	1175
C.3.2.4 CounterReset	1175
C.3.2.5 Error	1175
C.3.2.5.1 ErrorCode	1176
C.3.2.6 Event	1176
C.3.2.7 Milestone.....	1177
C.4 Milestone Values	1177
C.5 Input Tray and Output Bin Names	1179
Appendix D Supported Error Codes in JMF XJMF and Notification Elements	1183
Appendix E Color Adjustment Attribute Description and Usage ..	1187
E.1 Adjustment Using Direct Attributes	1187
E.2 Adjustment using ICC Profile Attributes	1188
E.3 Adjustment using an ICC Abstract Profile Attribute	1188
E.4 Adjustment using an ICC DeviceLink Profile Attribute	1188
Appendix F North American and Japanese Media Weight Explained ..	

1189	
F.1 North American Media Weight	1189
F.2 Japanese Media Weight	1190
Appendix G Media Sizes	1193
Appendix HMimeType and MimeTypeVersion Attributes	1197
Appendix I Generating strings with Format and Template	1203
Appendix J Pagination Catalog	1207
J.1 How to interpret the diagrams	1207
J.1.1 Legend	1207
J.1.2 Meaning of a Pagination Scheme	1208
J.1.3 Settings that Modify the Pagination Schemes.....	1208
J.1.3.1 BindingOrientation.....	1208
J.1.3.2 Binding and Jog Sides	1209
J.1.4 Getting a Specific Pagination Scheme.....	1209
J.1.4.1 Using the Settings to Find the Needed Scheme Transformation	1209
J.1.4.2 Scheme Transformations	1210
J.1.5 Examples	1212
J.1.5.1 Signature with Horizontal Binding Edges.....	1212
J.1.5.2 Signature with Vertical Binding Edges	1213
J.2 Pagination Diagrams	1214
Appendix K Resolving RunList/@Directory and FileSpec/@URL URI References	1255
K.1 Semantics of the RunList/@Directory Attribute	1255
Appendix L References	1257
Appendix M JDF XJDF/CIP4 Hole Pattern Catalog	1271
Appendix N FileSpec Attributes and Container Subelement	1281
N.1 Examples of Attribute Values of FileSpec	1281
N.2 Corresponding XML examples	1282
N.3 Additional examples showing Partitioning of FileSpec	1284
N.4 Example of an Intent Job Ticket with a doubly nested ZIP packaging file .	1290
N.5 AppOS and OSVersion Attributes	1291

Appendix O Examples	1293
O.1 Brief Example	1293
O.1.1 Before and After Processing	1293
O.2 Product JDF XJDF	1295
O.3 Spawning and Merging	1297
O.4 RunList	1304
O.5 Messages	1306
O.5.1 Simple KnownMessages.....	1306
O.5.2 Simple persistent channel.....	1307
O.5.3 JMF XJMF Pipe Messages	1308
O.5.3.1 Example Printer JDF XJDF	1308
O.5.3.2 Example Finisher JDF XJDF	1309
O.5.3.3 Initiation of PipePush sequence	1309
O.5.3.4 Continuation of PipePush sequence.....	1310
O.5.3.5 Paper Jam in finisher - PipePause	1312
O.5.3.6 Paper jam cleanup in finisher - PipePull.....	1312
O.5.3.7 Continued PipePush	1312
O.5.3.8 Paper jam in printer - PipePause	1314
O.5.3.9 Optional PipePause from the Finisher	1314
O.5.3.10 Optional PipePull from the Finisher	1314
O.5.3.11 Paper jam cleanup in printer - PipePush	1315
O.5.3.12 Job done - PipeClose.....	1315
O.6 Stripping	1316
O.7 DigitalDelivery Examples	1323
O.8 Automated Imposition	1330
Appendix P Deprecated Elements, JMF Messages, Processes and Resources	1337
P.1 Deprecated Structures of JDF Nodes and Jobs	1337
P.1.1 PlaceHolder Resource.....	1337
P.1.2 ResourceUpdate.....	1337
P.1.3 StatusPool	1338
P.1.3.1 PartStatus	1338
P.2 Life Cycle of JDF	1339
P.2.1 Case 5: Spawning and Merging of Independent Jobs.....	1339
P.3 JMF Messaging Elements	1341
P.3.1 Signal.....	1341
P.3.1.1 Trigger	1341
P.3.1.2 Added.....	1341
P.3.1.3 ChangedAttribute.....	1341

P.3.1.4 Removed	1342
P.3.2 Events.....	1342
P.3.2.1 NotificationDef	1343
P.3.3 KnownControllers	1343
P.3.3.1 ControllerFilter	1344
P.3.3.2 JDFController	1344
P.3.4 RepeatMessages.....	1345
P.3.4.1 MsgFilter	1345
P.3.5 NodeInfo	1346
P.3.5.1 NodeInfo Query	1346
P.3.5.1.1 NodeInfoQuParams	1347
P.3.5.2 NodeInfo Command	1347
P.3.5.2.1 NodeInfoCmdParams	1347
P.3.5.2.2 NodeInfoResp	1348
P.3.6 KnownJDFServices	1349
P.3.6.1 JDFService.....	1349
P.3.7 Occupation	1350
P.3.7.1 EmployeeDef	1350
P.3.7.2 Occupation.....	1351
P.3.8 Track.....	1352
P.3.8.1 TrackFilter	1352
P.3.8.2 TrackResult.....	1352
P.3.9 QueueEntryStatus	1353
P.3.9.1 QueueEntryDefList.....	1353
P.4 Deprecated Processes	1354
P.4.1 DBDocTemplateLayout	1354
P.4.2 DBTemplateMerging.....	1354
P.4.3 FormatConversion	1354
P.4.4 Ordering.....	1355
P.4.5 Packing.....	1355
P.4.6 FilmToPlateCopying	1356
P.4.7 PreflightAnalysis	1356
P.4.7.1 PreflightDetail	1357
P.4.7.2 PreflightInstance	1357
P.4.7.3 PreflightInstanceDetail	1358
P.4.8 PreflightInventory.....	1358
P.4.9 PreflightProfile	1359
P.4.9.1 PreflightConstraint.....	1360
P.4.10 Proofing	1360
P.4.11 SoftProofing.....	1361
P.4.12 IDPrinting.....	1362
P.4.13 AdhesiveBinding.....	1363
P.4.14 Dividing.....	1364

P.4.15 LongitudinalRibbonOperations	1364
P.4.16 Numbering	1364
P.4.17 SaddleStitching.....	1365
P.4.18 SideSewing.....	1365
P.5 Deprecated Intents	1366
P.5.1 BindingIntent Deprecated Subelements	1366
P.5.1.1 AdhesiveBinding	1366
P.5.1.2 BookCase.....	1366
P.5.2 DeliveryIntent Deprecated Subelements	1367
P.5.2.1 Pricing.....	1367
P.5.2.2 Payment	1367
P.5.2.3 CreditCard	1367
P.5.3 NumberingIntent.....	1368
P.5.3.1 NumberItem.....	1369
P.5.4 SizeIntent.....	1369
P.6 Deprecated Parameters	1370
P.6.1 AdhesiveBindingParams	1370
P.6.2 BoxFoldingParams Deprecated Subelements.....	1371
P.6.2.1 BoxApplication.....	1371
P.6.3 CustomerMessage	1372
P.6.4 DBMergeParams	1372
P.6.5 DBRules	1373
P.6.6 DBSchema	1373
P.6.7 DBSelection.....	1374
P.6.8 DividingParams	1374
P.6.9 FormatConversionParams.....	1374
P.6.10 IDPrintingParams	1375
P.6.10.1 Cover	1377
P.6.10.2 IDPFinishing.....	1378
P.6.10.3 IDPFolding	1379
P.6.10.4 IDPHoleMaking.....	1379
P.6.10.5 IDPLayout	1380
P.6.10.6 IDPStitching	1381
P.6.10.7 IDPTrimming.....	1383
P.6.10.8 ImageShift	1383
P.6.10.9 JobSheet.....	1384
P.6.11 Layout Deprecated Subelement	1386
P.6.11.1 Signature	1386
P.6.12 LongitudinalRibbonOperationParams.....	1387
P.6.12.1 LROperation	1387
P.6.12.2 LongFold	1388
P.6.12.3 LongGlue	1388

P.6.12.4 LongPerforate	1388
P.6.12.5 LongSlit	1388
P.6.13 MediaSource	1388
P.6.14 NumberingParams	1389
P.6.15 NumberingParam	1389
P.6.16 OrderingParams	1390
P.6.17 PackingParams	1390
P.6.18 PlaceHolderResource	1391
P.6.19 PlateCopyParams	1392
P.6.20 ProofingParams	1392
P.6.21 RunList Deprecated Subelements	1394
P.6.21.1 DynamicInput	1394
P.6.22 SaddleStitchingParams	1394
P.6.23 Sheet	1395
P.6.24 SideSewingParams	1396
P.6.25 Surface	1397
Appendix Q List of Figures	1399
Appendix R List of Tables	1403
Appendix S List of Examples	1427

Table of Contents

-
- E

JMF

Resource

Layout

Product Intent

ApprovalDetails

QueueSubmissionParams

DeviceNSpace

Contact

MarkObject PlacedObject ContentObject

DynamicField

RunList**FileSpec****ColorantControl****ExposedMedia**

ResourceSet

Component**Component****Media**

Media

AmountPool

BundleItem

Device**Device****Ink**

Address

BinderySignature**ProductionPath**

DeviceInfo

ResourceInfo

AuditPool

MiscConsumable**Tool****UsageCounter****DensityMeasuringField**

ColorantAlias
GeneralID

Contentp77

Chapter 1 Introduction

This document defines the technical specification for the Job Definition Format (**XJDF**) and its counterpart, the Job Messaging Format (**XJMF**).

XJDF is a technology that allows systems from many different vendors to interoperate in automated workflows. While technically it is an XML software specification, it is more importantly a means to connect multiple vendor solutions to a workflow solution for automation.

XJDF is the first major version update of JDF. It is such a major update that we decided to provide a new root node name: **XJDF**. Whereas the minor revisions were at least nominally backwards compatible, **XJDF** is a major redesign that takes more than a decade of experience into account.

This document is intended for use by programmers and systems integrators. It provides both the syntactical requirements for the elements and attributes of **XJDF** and **XJMF** as well as requirements for devices and controllers to enact upon the data. In this first chapter, we present the concept of **XJDF**, and its relationship to JDF and other industry standards.

1.1 Background on XJDF

XJDF is an extensible, XML-based data interchange format built upon more than 15 years of experience with JDF [JDF15].

XJDF is an interchange data format that can be used by a system of controllers, devices and MIS, which together produce printed products. It provides the means to describe print jobs in terms of the products eventually to be created, as well as in terms of the work steps needed to create those products. **XJDF** provides a syntax to explicitly specify the details of processes, which might be specific to the devices that execute the processes.

XJDF is aligned with a communication format known as the Job Messaging Format or **XJMF**. **XJMF** provides the means for production components of an **XJDF** workflow to communicate with controllers such as MIS. It gives MIS and other controllers the ability to receive information from devices or other controllers about the status of jobs and devices. **XJDF** and **XJMF** are maintained and developed by CIP4 (<http://www.cip4.org>).

1.2 Design Criteria for XJDF

The major conceptual change is that **XJDF** no longer attempts to model the entire job as one large “job ticket” but rather specifies an interchange format between 2 applications that are assumed to have an internal data model that is not necessarily based on **XJDF**. Thus each **XJDF** ticket specifies a single transaction between two parties. A single job may be modelled as one or more **XJDF** transactions.

The following criteria were taken into account in this redesign:

- **XJDF** should be simple to use.
- The number of methods to describe similar traits should be as limited as possible, ideally one.
- **XJDF** should be compatible with developments of XML enabling tools as “out of the box” as possible.
 - Simple Xpath expression to reference **XJDF**. traits.
 - Direct use of ID-IDREF pairs for referencing distributed data within an XML document.
 - Use of XML schema rather than proprietary data structures to describe device capabilities.
- The semantics of JDF 1.x should be retained and mapping between JDF 1.x and **XJDF** should be simple.
- Change orders (Modifications of submitted jobs) should be easy to describe

These requirements lead to some significant modifications that are not syntactically backwards compatible, but can easily be converted using a JDF 1.x aware middleware.

1.2.1 Simplify and reduce variations

JDF 1.x allowed shorthands for some simple cases. What seemed reasonable actually made things more complex, since both shorthand and the long version had to be implemented. For instance, amount related attributes could be found either directly in a ResourceLink or in a ResourceLink/AmountPool/PartAmount. XJDF removes much of this variability.

1.2.1.1 Reduce the barrier of entry

Simple tasks should be simple to describe. In the simple (non-partitioned job metadata cases) the XJDF should be describable as a series of simple XPaths.

1.2.1.2 Replace abstract data types with explicit elements and children

Abstract elements are more concise to write, but inherited traits also tend to be overlooked by newcomers to a specification. If elements are designed to be final with sub-elements, each specification entry can be found by searching for the explicit element name.

1.2.1.3 Remove ResourceLinks

The Process / Resource model has been conceptually retained. But since there is only one XJDF element per XJDF transaction, reuse of resources is no longer an issue and ResourceLinks have been merged with their respective Resources. Thus data that belongs together is also stored in the same region of the XML.

For each refElement (i.e., choice of resourceRef or inline element), exactly one choice was made. Thus the variability is reduced and implementation is simplified.

1.2.2 Enable dynamic changes

The monolithic model of JDF 1.x lent itself well to a plan and execute philosophy but had its limitations when changes were made after a job had been submitted. Since a Job may be modelled as a set of transactions in XJDF, the idea of multiple transactions and thus also job changes is inherently built into the standard. The simplest method of initiating a change transaction is to send an XJDF that contains only the modified values. Only the explicitly stated values will then be modified.

1.2.2.1 Remove schema defaults.

Since XJDF allows submission of incomplete XJDF tickets and also updates to preexisting tickets, all attributes except very few that are required for bookkeeping (e.g., @ID) have been made optional and all schema defaults have been removed.

1.2.3 Retain the semantic structures

A lot of work was put into the definition of individual messages, processes and resources. The detailed semantics of JMF messages and resources have been retained. Thus detailed element and attribute names and their definitions have been retained. Thus translation between JDF 1.x and XJDF is straight forward. All deprecated traits have been removed completely.

1.2.4 Remove implementation specific details

JDF 1.x exposes many implementation details that are not necessarily easily obtained by the writers of JDF. XJDF is designed as a pure interface specification that encapsulates internal data as much as possible.

1.2.4.1 Spawning and Merging

Since XJDF is only an interface, the specification of serializing from the internal data model and deserializing to the internal data model is outside the scope of this specification and has been removed.

1.2.5 Enhance Compatibility with standard XML and XML Tools

XML and XML related tools and technologies such as XPath, XSL transforms, Schema, class generators etc. have evolved and matured significantly since the turn of the century. Some of the choices in JDF 1.x, although compliant with XML have proved difficult to implement using standard tools.

1.2.6 Partitioning and Inheritance

The general concept of partitioning (i.e., the notion of resource sets with multiple individual parts) is retained but the encoding has been simplified. While inheritance of elements and attributes in partitioned resources potentially can reduce redundancy of the data encoded in XJDF, it also greatly increases the flexibility and variability of specifying similar data. This causes potential for reader/writer mismatch. Inheritance and the corresponding definition of cardinality (e.g., “SHALL occur somewhere in the inherited hierarchy”) is also difficult to encode as XPath or in an XML schema. XJDF therefore removes inheritance at the cost of redundant specification of traits in partitioned resources. For details of partitioning, see ##ref partitioning.

1.2.6.1 Removal of Partition SignatureName

@SignatureName in JDF was used to describe a set of multiple printed sheets, which is contrary to the usage of Signature in traditional printing. Since most systems refer directly to sheets, the *@SignatureName* partition key was removed.

1.2.7 Device Capabilities

JDF provided proprietary methods to describe device limitations. XML schema is a standard technology that is also designed for this purpose albeit with some limitations such as the lack of a mechanism to describe constraints dependencies. Nonetheless we decided to define device limitations using XML schema in order to make use of the existing tool base for XML schema.

1.3 Document References

Throughout this specification references to other documents are indicated by short symbolic names inside square brackets (e.g., [ICC.1]). Implementers ought to read and conform to such referenced documents when implementing a part of this specification with such a reference. The reader is directed to Appendix L, “References” on page 1257 to find the complete set of **XJDF** references and the full title, date, source and availability of all such references. In addition, this specification assumes that the reader has a basic awareness of or access to, the following documents.

Table 1-1: Basic References (Sheet 1 of 2)

Term	Definition
[XML]	<i>Extensible Markup Language (XML) 1.0 (Fifth Edition)</i> <i>Version (W3C Recommendation of 26 November 2008)</i> Date: 26 November 2008 Produced by: World Wide Web Consortium (W3C) Available at: http://www.w3.org/TR/2008/REC-xml-20081126/
[XMLNS]	<i>Namespaces in XML 1.0 (Third Edition)</i> <i>Version (W3C Recommendation of 8 December 2009)</i> Date: 8 December 2009 Produced by: World Wide Web Consortium (W3C) Available at: http://www.w3.org/TR/2009/REC-xml-names-20091208/
[XPath]	<i>XML Path Language (XPath) 2.0 (Second Edition)</i> <i>Version W3C Recommendation 14 December 2010</i> Date: 14 December 2010 Produced by: World Wide Web Consortium (W3C) Available at: http://www.w3.org/TR/2010/REC-xpath20-20101214/

Table 1-1: Basic References (Sheet 2 of 2)

Term	Definition
[XMLSchema]	<p><i>XML Schema Part 0+1+2: Primer, Structures and Datatypes Version (W3C Recommendation of 28 Oct 2004)</i></p> <p>Date: 28 October 2004 Produced by: World Wide Web Consortium (W3C) XML Schema working group Available at: http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/, http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/ and http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/.</p>

1.4 Conventions Used in This Specification

This section contains conventions and notations used within this document.

1.4.1 Text Styles

The following text styles are used to identify the components of an **XJDF Job**.

- Elements are written in sans serif. Examples are **Comment**, **BundleItem** and **Resource** Elements. Attributes are written in italic sans serif. Examples are **@Status**, **@ResourceID** and **@ID**.
- Resources are written in bold sans serif. Examples are **Layout** and **ExposedMedia**.
- Process types are written in bold-italic sans serif. Examples are **ColorSpaceConversion** and **Rendering**.
- Enumerative and Boolean values of Attributes are written in italics. Examples are **"true"**, **"Waiting"**, **"Completed"** and **"Stopped"**.
- Standard bold text is used for the following purposes
 - to highlight glossary items. Examples are **Device**, **Element** and **Job**.
 - to highlight defined items inside a table. An example is the data type **NMTOKEN** in the table in Section 1.6, “Data Structures”.
 - to highlight definitions of local terms. These are terms that are of local importance for a certain chapter or some sections inside a chapter. An example is a **Logical Sheet** in Section 6.4.12.2, “Variables for Automated Imposition” on page 370
- For the benefit of those who are reading this document in PDF or online, cross-reference links are denoted by gray text. Examples are Chapter 4, “Life Cycle of XJDF” on page 183 and Section 1.4, “Conventions Used in This Specification” on page 4. To follow a link, click the highlighted text.
- Also for the benefit of online readers, external hyperlinks are graphically designated. An example is <http://www.cip4.org>. To follow a link, click the highlighted text.

1.4.2 XPath Notation Used in this Specification

A simple subset of the XPath Language [XPath] is used throughout this specification in the description of an Element, Attribute or value to identify other Elements, Attributes and/or values. XPath gets its name from its use of a path notation (as in URLs) for navigating through the hierarchical structure of an XML document. The simple subset of XPath used is:

- Element Subelement hierarchy is indicated by a slash (e.g., “Element/Element”).
- Element Attribute hierarchy is indicated by a slash and an at (@) symbol (ex., “Element/@Attribute”), and
- The text styles above in Section 1.3.1 are used to indicate whether an Element is a Resource, Process or other Element, or if the subject is an Attribute or a value (ex., enumeration, string, etc.).
- Paths beginning with a single slash: “/” indicate root Elements: (ex. /XJDF indicates the root Element).
- Paths beginning with a double slash “//” indicate Attributes or Elements that reside in any parent.

- Paths containing square brackets that enclose a predicate describe an Element that is restricted by the predicate. This document uses three types of predicates as described in the next 3 items:

E[@A = V] – the XPath specifies an Element **E** whose Attribute **A** has the value **V** (e.g., **Component[@ComponentType = "FinalProduct"]** specifies a **Component** whose **@ComponentType** has the value of **"FinalProduct"**). The predicate can be used outside the context of an Element (e.g., **@ComponentType = "FinalProduct"** (or without the "@": **ComponentType = "FinalProduct"**) means that the **@ComponentType** value equals **"FinalProduct"**).

E[contains(@A = V)] – the XPath specifies an Element **E** whose Attribute **A** has some value that contains **V**, **A**'s value is either an enumerations or NMOKENS and **V** is an enumeration or NMOKEN. For example, **Contact[contains(@ContactTypes = "Delivery")]/Address** specifies the **Address** of a **Contact** whose **@ContactTypes** value contains **"Delivery"** and possibly other NMOKEN values. The predicate can be used outside the context of an Element (e.g., **contains(@ContactTypes = "Delivery"** means that **@ContactTypes** value contains **"Delivery"**).

E[@A] – the XPath specifies an Element **E** in which Attribute **A** is present (e.g., **Layout[@Side]** specifies a **Layout** in which the **@Side** Attribute is present).

Example 1-1: XPath Expression

The XPath expression:

- Layout/MarkObject/DynamicField/@Format = "Replacement Text for %s and %s go in here at %s on %s"...**

Means:

- The value **"Replacement Text for %s and %s go in here at %s on %s"** of the **@Format** Attribute of the **DynamicField** Subelement of the **MarkObject** Element of the **Layout** Resource Element.

Locally, (and within context), just the basic Attribute dependency is noted (for instance — **DynamicField/@Format**) where the discussion occurs within the section describing the Element (e.g., **DynamicField** in our example) Elements or the Element's immediate parent (e.g., **MarkObject** in our example).

1.4.3 Modification Notes

To help the reader familiar with earlier versions of **XJDF**, future versions of this specification will indicate additions, deprecations and clarifications using the callouts described in Table 1-2. Modifications with respect to JDF 1.x are NOT called out explicitly. Please note that not all changes are identified with modified callout flags. When modification occurs in multiple versions, only the most recent version is indicated. A few changes have been made globally and are explained in the body of the document and only significant changes have been flagged with callouts, as determined by CIP4 Working Groups.

Table 1-2: Modification Notes

Example	Callout Meaning
New in XJDF 2.x	New sections, Attributes/Elements and Attribute Values.
Deprecated in XJDF 2.x	Deprecated sections, Attributes/Elements and Attribute Values. Usually there is a deprecation note describing the mechanism that replaces the deprecated item.
Modified in XJDF 2.x	Changed syntax or semantics of sections or Attributes/Elements. Might include clarification as well. Frequently there is a modification note describing the change.

1.4.3.1 Location of Modification Notes

A callout occurs after one of the following document elements.

- **Section head:** applies to entire section and the contained table (if any).
- **Attribute/Element name:** applies to entire row for the designated Attribute/Element.
- **Attribute value:** applies to Attribute value.

1.4.4 Specification of Cardinality

The cardinality of XJDF Attributes and Elements is expressed using a simple Extended Backus-Naur Form (EBNF) notation.

The symbol **T** in the table below represents an Attribute or Element. The symbol **T** consists of either a single name, such as “**RunList**” or an Element name followed by a parenthesized name, such as “**RunList (Document)**”. The name in parentheses “Document” identifies a particular Element instance when several of the same type exist in some context. For further details, see Section 6.2, “Process Template” and Section 1.4.6, “Template for Tables that Describe Elements”.

Table 1-3: Cardinality Symbols

Notation	Description
T	T SHALL occur exactly once and represents an Attribute or Element.
T ?	T is OPTIONAL or is REQUIRED only in the circumstances explained in the description field. T represents an Attribute or Element. If T is an Attribute, a default that is specified in the description will not be inserted into the XML by a schema aware parser if no value is explicitly specified.
T +	T occurs one or more times, and represents an Element.
T *	T occurs zero or more times, and represents an Element.

1.4.5 Template for Narrative Description of Resources

Each section for a Resource begins with a brief narrative description of the Resource. Following that is a list containing details about the properties of the Resource, as shown below.

Resource Properties

Resource referenced by: List of Elements that MAY contain references to Elements (IDREF or IDREFS Attributes) of this type.

Input of Processes: List of **XJDF/@Types** values that use the Resource as an Input Resource

Output of Processes: List of **XJDF/@Types** values that create the Resource as an Output Resource

After the list describing the Resource properties, each section contains tables that outline the structure of each Resource and, when applicable, the Subelement information that pertains to the Resource structure. The first column contains the name of the Attribute or Element. In some cases, a Resource will contain multiple Elements of the same type. If this is the case, the Element name is listed as often as it appears, along with a term in parentheses that identifies the occurrence. For an example, see Section 8.37, “Resource EndSheetGluingParams”. The following sections provide templates of the tables.

1.4.6 Template for Tables that Describe Elements

Resources and Elements are defined by their attributes and sub-elements.

Note: for tables that describe Resources or Elements:

- the *italicized* text describes the actual text that would be in its place in an actual Resource definition
- *Cardinality* in the Name column refers to a cardinality symbol, which is either empty or consists of a symbol, such as “?”. Examples described by the Name column include: “**Media***” and “**Component ?**”. For further details, see Section 1.4.4, “Specification of Cardinality”.

- The text following a “**Note:**” in a table field gives further information about the specified table row.

Table 1-4: Template for Element Descriptions

Name	Data Type	Description
<i>Attribute-Name Cardinality</i>	<i>Attribute-data-type</i>	<i>Information about the Attribute.</i>
<i>Attribute-Name Cardinality</i>	IDREF/IDREFS	If the data type of an attribute is IDREF or IDREFS and the element is a Resource, then the referenced ID or set of IDs SHALL be specified in the respective container Resource.
<i>Element-Name Cardinality</i>	element	<i>Information about the Element.</i> Note: the “element” data type means that the specified Element SHALL be an in-line Subelement within the Resource.
FileSpec (<i>someValue</i>) <i>Cardinality</i>	element	<i>Information about the FileSpec Resource</i> Note: FileSpec / <i>@ResourceUsage</i> SHALL match the “ <i>someValue</i> ” value specified in the parentheses. When a Resource potentially contains multiple FileSpec children, the value of FileSpec / <i>@ResourceUsage</i> distinguishes the FileSpec Resources.

1.5 Glossary

The following terms are defined as they are used throughout this specification. For more detail on **Job** and workflow components, see Section 2.1, “System Components”.

Table 1-5: Glossary (Sheet 1 of 5)

Term	Definition
Allowed values are	The phrase “ Allowed values are: ” precedes the complete (closed) list of values for an Attribute whose values are enumeration or enumerations.
Allowed values are from	The phrase “ Allowed values are from: ” precedes a reference to the values. The reference may be an XPath to a value or a reference to a table of Attribute Values . The referenced values are as complete (closed) as the reference says they are.
Attribute	An XML syntactic construct describing an unstructured characteristic of an Element . See [XML] for details.
Attribute Value	The value of an Attribute .
Binding Side	The binding side of a Signature is defined as the last fold.
Command Message	A Command Message is a XJMF Message that requests its recipient to change its state.
Controller	The component of an XJDF -based workflow that initiates Devices , routes XJDF , and communicates status information. MIS is an example of a top level controller.
Deprecated	Indicates that an XJDF Element is being phased out of XJDF usually in favor of newer XJDF Element(s) . It is RECOMMENDED that an Agent not include such an XJDF Element in an XJDF instance. Such an indicated XJDF Element might be removed from a future version of the XJDF specification. XJDF Consumers SHOULD only Support such XJDF Elements for backward compatibility with previous versions of XJDF . Deprecated items are flagged with Deprecated in XJDF 2.x in this specification.

Table 1-5: Glossary (Sheet 2 of 5)

Term	Definition
Device	The component of an XJDF workflow part that interprets XJDF and executes the instructions. If a Device controls a Machine , it does so in a proprietary manner. For details, see Section 2.1.4.2, “Devices” about devices in workflow components.
Document Set	A set of Instance Documents presumed to be related.
Element	An XML-based syntactic construct describing structured data in XJDF .
Enumeration	A data type that represents a closed set of values, which are specified in this document.
Finished Page	A page of a final product that normally has no folds inside. The folds of the finished product for packaging (e.g., folding letters into an envelope) or Z-fold of an oversized book, have no effect on the Finished Page definition. A Sheet of paper with no fold inside consists of two Finished Pages (“recto” and “verso” or front and back side). If there are folds seen in a Sheet in the final product, the number of Finished Pages of one Sheet is given by $2*(X+1)*(Y+1)$, where X denotes the number of folds in X direction and Y denotes the number of folds in Y direction, each seen in the completely opened Sheet . Examples: One Sheet in a book has two Finished Pages , one front, one back; a brochure with one fold inside has four Finished Pages .
Form	A collection of imposed (ordered) Finished Pages set for printing or imaging to plate or film. See also Signature .
Gear Side	Gear Side is the side of a Machine , where the drives and gear are mounted. Gear Side is opposite to Operating Side .
Gray Box	A Gray Box is an incomplete specification of a process. Resource is required for actual production MAY but NEED NOT be complete.
Input Resource	A Resource that is an input to a Process . See Resource .
Instance Document	A document that is part of the output of a Job . This generally refers to personalized printing Jobs . Each of the individual documents produced from the same input template is referred to as an Instance Document . For example, in a credit card statement run, each statement is an Instance Document .
Intent	An Intent is an Element that defines the details of Products to be produced without defining the process to produce them. Intent elements typically describe aspects of the end-customer view of a printed product.
Job	A collection of one or more Job parts. Note that a Job does not necessarily have an XML representation.
Job Part	A granular task that is represented by a single XJDF .
Jog Edge	The jog edge of a Signature is defined as the last fold that is perpendicular to the Binding Side . If there is no fold that is perpendicular to the Binding Side , then the jog edge is the edge on the bottom when the folded signature is not flipped after the final fold.
Link	A pointer to information that is located elsewhere in a XJDF document or that is located in another document.
Machine	The part of a device that does not know XJDF and is controlled by a XJDF Device in a proprietary manner.
Message	The XML element that Devices and Controller use to exchange queries, commands, responses, etc. among themselves using HTTP as the underlying protocol and XJMF as the XML format. See XJMF and Message Family .

Table 1-5: Glossary (Sheet 3 of 5)

Term	Definition
Message Family	A Message Family is a set of Messages . The 4 Message Families are Query Message , Command Message , Response Message and Signal Message .
MIS	Management Information System. The functional part of a XJDF workflow that oversees all Processes and communication between system components and system control. MIS is assumed to be a role rather than an individual application. A single application may fulfill various roles of an MIS and various roles of an MIS may be implemented by multiple applications. Typical MIS roles include estimation, costing, scheduling, process planning and invoicing.
Name	The name of a Resource and is the value of ResourceSet/@Name (e.g., ColorantControl or ExposedMedia).
NamedFeature	The term NamedFeature describes a value that is identified by a name using a JDF/GeneralID[@DataType = "NamedFeature"] . The value is specified by GeneralID/@IDValue , and the name is specified by GeneralID/@IDUsage . For example, the GeneralID IDUsage="a1" IDValue="a2 b2" DataType="NamedFeature" is a NamedFeature with a value "a2 b2" that is identified by the name "a1". GeneralID/@IDValue is defined as a string. Note that blanks are allowed in the value.
Operating Side	Operating Side is the side of a Machine , where the operator works. Operating Side is opposite to Gear Side .
Output Resource	A Resource that is an output from a Process . See Resource .
Page	When Page occurs by itself, not in the context of Finished Page or Reader Page , it means "Finished Page."
Partition	Any Resource that belongs to a set of Resources but describes one part of the Resource is referred to as a partition. Resources within ResourceSets are partitions.
Partition Key	A Partition Key is an Attribute that can identify a Partition . It selects the specific Resource from its parent ResourceSet . Partition Keys are Attributes within a Part Element . See Section 3.6.2.2, Selecting a Partition.
PDL	Page Description Language. A generic term for any language that describes pages which might be printed. Examples are PDF®, PostScript® or PCL®.
Phase	A Phase is a distinct part of a Workstep such as setup, production or cleanup.
Process	An individual step in the workflow.
Product Intent	Describes the end result that a customer is requesting. See Section 7, Product Intents.
Query Message	A Query Message is an XJMF Message that requests its recipient to provide information, but not change its state.
Queue	A Queue is a representation of a Device that receives, manipulates and stores multiple Queue Entry items for execution.
Queue Entry	A Queue Entry is a the representation of an individual XJDF process in a Queue . A Queue Entry MAY be comprised of multiple Workstep processes.
Reader Page	A logical page as perceived by a reader, for example one RunList entry. One Reader Page might span more than one Finished Page (e.g., a centerfold). One Finished Page might contain contents defined by multiple Reader Pages (e.g., NUp imposition). Reader Pages are defined independently of Finished Pages .
Receiver	Device or Controller that responds to an XJMF request, e.g. by processing an HTTP post request. The sender is an HTTP server.
Resource	A Resource is an Element that defines a physical entity that is modified or used by an XJDF Node . Examples include paper, plates and ink. See Section 8.1, "Resource".

Table 1-5: Glossary (Sheet 4 of 5)

Term	Definition
Response Message	A Response Message is an XJMF Message that functions as a synchronous response to a Command Message or Query Message.
Roll	A Roll is media that is mainly used in connection with Web Printing. In British English the name “reel” for “roll” is in widespread use. Roll is used as synonym of reel. Also, see the term Web in this glossary.
Sender	Device or Controller that initiates an XJMF exchange, e.g. by sending an HTTP post request. The sender is an HTTP client.
Sheet	<p>The printer’s Roll of paper or paper cut for press size, with “recto” and “verso” forms for identification of orientation through the press (facing up versus facing down at the feeder or off the Roll).</p> <p>Sheets are press sheets which may be comprised of multiple folding Signatures and might also have “recto” and “verso” forms for identification of orientation through the press (facing up versus facing down at the feeder or off the roll).</p> <p>The term “cut sheet” refers to an individual Sheet, typically in a phrase, such as “separately cut Sheets of an opaque material”. The term “Sheet-Fed” is used to describe a press that consumes cut Sheets, typically in the phrase “Sheet-Fed Press”.</p>
Signal Message	A Signal Message is an XJMF Message that is sent asynchronously when some event occurs.
Signature	A Signature is a Sheet that is folded or will be folded. A press MAY contain multiple signatures.
Slave Controller	The component of a XJDF workflow that accepts XJDF as a Device from other Controllers and/or Slave Controllers and sends XJDF to other Slave Controllers and/or Devices .
Subelement	A child Element of some other Element
Support	An XJDF Consumer Supports a XJDF syntactic construct (Processes , Resources , Elements , Attributes and Attribute Values) if the XJDF Consumer performs the action defined in this specification for the XJDF construct when consuming an XJDF instance that includes the XJDF syntactic construct. If the Machine that a Device is representing Supports a feature which is represented by an XJDF construct, then the Device SHOULD Support that XJDF syntactic construct.
Surface	A single side of a Sheet .
Tag	A syntactic XML construct that marks the start or end of an Element .
Unique	<p>The word “Unique” without further scope details means “Unique within the job, message or file depending on the context”.</p> <p>Possible scopes are:</p> <ul style="list-style-type: none"> Unique within the machine (e.g., ProductionPath/@ProductionPathID). Unique within the workflow – This covers the whole scope at one installation (e.g., StatusQuParams/@JobID, FileSpec/@UID, Device/@DeviceID). Unique within the company – identification of a company or contact SHALL be unique within the company’s database because it can be used on multi tenant systems like web approval) (e.g., Contact[(@ContactTypes, "Approver")]/@ExternalID).
Values include	The phrase “ Values include: ” precedes an open list of values for an Attribute whose values are NMTOKEN , NMTOKENS or string. The list includes recommended values but does not include preclude potential vendor or customer extensions.
Values include those from	The phrase “ Values include those from: ” precedes a reference to the open list of values. The reference may be an XPath to a value or a table of Attribute Values . The referenced values do not include potential vendor or customer extensions.

Table 1-5: Glossary (Sheet 5 of 5)

Term	Definition
Web	A Web is media that comes from a Roll and is mainly used in connection with Web Printing . This specification uses the word “Web-Fed” instead of “roll-fed”. It uses the phrase “Web Printing” and “Web Press” to describe printing presses that consumes media that comes from a Roll .
Work Center	An organizational unit, such as a department or a subcontracting company, that can accomplish a task.
Workstep	A Workstep is an individual XJDF process that can be processed on a single device in one pass. A Workstep is comprised of one or multiple Phase parts such as setup, production or cleanup.
XJDF	Job Definition Format, version 2. The overall name of this specification is XJDF. There is also a JDF Element , which is a top-level Element within XJDF that encompasses a XJDF Node (see above).
XJDF Consumer	A Device or Controller that consumes XJDF instances.
XJDF Node	The XJDF Element type detailing the Resources and Process specification needed to produce a final or intermediate product or Resource .
XJMF	Job Messaging Format. Transfers information between Controllers and Devices . See Chapter 5, “XJMF Messaging with the Job Messaging Format” on page 207.
XJMF Message	An XJMF Message is synonymous with Message . See Message .

1.5.1 Conformance Terminology

The words “SHALL”, “SHALL NOT”, “REQUIRED”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, “NEED NOT” and “OPTIONAL” are used in this specification to define a requirement for the indicated XJDF Consumer as follows.

Table 1-6: Conformance Terminology (Sheet 1 of 2)

Term	Meaning
SHALL or REQUIRED	Means that the definition is an absolute requirement of the specification.
SHALL NOT	Means that the definition is an absolute prohibition of the specification.
SHOULD or RECOMMENDED	Means that there might exist valid reasons in particular circumstances for an implementer to ignore a particular item, but the implementer SHALL fully understand the implications and carefully weigh the alternatives before choosing a different course.
SHOULD NOT or NOT RECOMMENDED	Means that there might exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the implementer should fully understand the implications and then carefully weigh the alternatives before implementing any behavior described with this label.

Table 1-6: Conformance Terminology (Sheet 2 of 2)

Term	Meaning
MAY, NEED NOT or OPTIONAL	Means that an item is truly optional. See Section 1.4.4, “Specification of Cardinality”. For features that are optional for an XJDF Consumer to Support , one vendor might choose to Support such an item because a particular marketplace requires it or because the vendor feels that it enhances the product, while another vendor might omit Support of that item. Similarly, one vendor of an Agent might choose to supply such an item in an XJDF instance, while another vendor might omit the same item in an XJDF instance. A XJDF Consumer implementation which does not include Support of a particular option (Element or Attribute) SHALL be prepared to interoperate with an Agent implementation which does supply the option, though with reduced functionality. In the same vein, a XJDF Consumer implementation which does include Support for a particular option SHALL be prepared to interoperate with an Agent implementation which does not supply the option in the XJDF instance.

1.5.2 Conformance Requirements for XJDF Entities

The subsections of this section define the general conformance requirements for the XJDF entities: 1) **Attributes** and **Attribute Values**, 2) **Resources**, and 3) **Processes**.

1.5.2.1 Conformance Requirements for Support of Attributes and Attribute Values

If an **XJDF Consumer** supports an Attribute, it SHALL support all of the values that this specification indicates are REQUIRED for an **XJDF Consumer** to support (whether or not the Attribute is REQUIRED for the **Agent** to supply in that context). If this specification is silent on which values are REQUIRED for support of an Attribute, then the **XJDF Consumer** SHALL support at least one value in order to claim support for the Attribute.

Attributes that are OPTIONAL for an **Agent** to include in an **XJDF** instance are indicated by a “?” character following the Attribute name as indicated in Section 1.4.4, “Specification of Cardinality”.

1.5.2.2 Conformance Requirements for Support of Elements

If an **XJDF Consumer** supports an Element, it

- 1 SHALL support all of the Attributes (see Section 1.5.2.1) defined for that Element that an **Agent** is REQUIRED to include in the Element instance Attributes with either no marks or a “+” as defined in Section 1.4.4, and

1.5.3 Interoperability Conformance Specifications

Interoperability Conformance Specifications (i.e., ICS documents) are developed by CIP4 working committees. They establish the minimum **XJDF** support requirements for Devices of a common class, including expected behavior. An ICS document can subset **XJDF** but can not expand upon **XJDF**. For instance, an ICS that covers desktop printers can either omit or prohibit all of the postpress Processes related to case binding. ICS documents can also establish minimum **XJMF** support requirements for a class of Devices.

Once published, ICS documents will form the basis for testing and certification by CIP4-sanctioned facilities. **XJDF**-enabled products that pass these tests will be deemed “**XJDF Certified**” to conform to an identified level of one or more ICS documents and will be permitted upon certification to use a “**XJDF Certified**” logo in connection with certified **XJDF**-enabled products.

The development of ICS documents are done in parallel, but not in synchronization, with the development of editions of the **XJDF** specification (e.g., an ICS is related to a specific edition of the **XJDF** Specification, but might be released at a later date). Once approved, all published ICS documents will be available at http://www.cip4.org/document_archive/ics.php.

1.6 Data Structures

The following table describes the data structures as they are used in this specification. Unless stated otherwise, this specification uses XML data types as defined by [XMLSchema]. For more details on **XJDF** Schema and data types, see Appendix A, “Encoding” on page 1143.

In JDF, some data types have been enhanced to include unbounded values by defining the explicit tokens "*INF*" and "*-INF*". For instance, the IntegerRange "*0 INF*" specifies all positive integers including 0. A range value consists of one or more pairs of values. Each pair of values is a range. If the two values in a pair are the same, the pair represents a single value. Each value is separated by space character.

Table 1-7: XJDF Data Types (Sheet 1 of 3)

Data Type	Description
Anchor	Describe the 9 anchor points of a rectangle. See Table A-1, “Anchor Enumeration Values” on page 1154 for a list values.
boolean	Binary-valued logic: (true false).
CMYKColor	Represents a CMYK color specification.
date	Represents a time period that starts at midnight of a specified day and lasts for 24 hours.
dateTime	Represents a specific instant of time. It SHALL be a UTC time or a local time that includes the time zone.
DateTimeRange	Two <i>dateTime</i> values separated by a space character that defines the closed interval of the two. TimeRange corresponds semantically to the time interval (two time instants separated by a slash) defined in [ISO8601:2004].
DateTimeRangeList	Whitespace-separated list of <i>DateTimeRange</i> values.
double	Corresponds to [IEEE754] double-precision, 64-bit floating point type, including special tokens INF and -INF. This corresponds to the standard XML double with NaN removed. For details, see [XMLSchema].
DoubleList	Whitespace-separated list of <i>double</i> values.
duration	Represents a duration of time.
DurationRange	Two duration values separated by whitespace. Describes a range of time durations. More specifically, it describes a time span that has a relative start and end.
DurationRangeList	Whitespace-separated list of <i>DurationRange</i> values.
element	Structured data. The specific data type is defined by the Element name.
enumeration	Limited set of <i>NMOKEN</i> values (see below).
enumerations	Whitespace-separated list of <i>enumeration</i> values.
gYearMonth	Represents a specific Gregorian month in a specific Gregorian year.
hexBinary	Represents arbitrary hex encoded binary data.
hexBinaryList	Whitespace-separated list of <i>hexBinary</i> values.
ID	Unique identifier as defined by [XML] (see Section 1.3, “Document References”). SHALL be unique within the scope of the XJDF document.
IDREF	Reference to an element holding the unique identifier as defined by [XML Specification 1.0].
IDREFS	List of references (IDREF values) separated by white spaces as defined by [XML].

Table 1-7: XJDF Data Types (Sheet 2 of 3)

Data Type	Description
integer	Represents numerical integer values, including the special tokens INF and -INF. This corresponds to the standard XML integer with INF and -INF added. Values greater than +/-2**31 are not expected to occur for this data type. For details, see [XMLSchema].
IntegerList	Whitespace-separated list of <i>integer</i> values.
IntegerRange	Two <i>integer</i> values separated by a space character that define a closed interval.
IntegerRangeList	Whitespace-separated list of <i>integer</i> values and <i>IntegerRange</i> values.
JDFJMFileVersion	Version label of an XJDF or JMF %%% instance. XJDF %%%% %%% See Table A-2, “JDFJMFileVersion Enumeration Values” on page 1154 for a list values.
JDFJMFDVersions	Whitespace separated list of <i>JDFJMFileVersion.values</i>
LabColor	Represents a Lab color specification.
language	Represents a language and country code (for example, en-US) for a natural language. Values SHALL conform to [RFC1766].
languages	Whitespace-separated list of <i>language</i> values.
LongInteger	Represents numerical integer values, including the special tokens INF and -INF. This corresponds to the standard XML integer with INF and -INF added. Values greater than +/-2**31 are expected to occur for this data type. For details, see [XMLSchema].
matrix	Whitespace-separated list of six doubles representing a coordinate transformation matrix.
NamedColor	Represents a color definition by name. See http://www.w3.org/TR/html4/types.html#h-6.5
NameRange	Two <i>NMOKEN</i> values separated by a space character that define an interval of <i>NMOKEN</i> values.
NameRangeList	Whitespace-separated list of <i>NameRange</i> values.
NMOKEN	A continuous sequence of special characters as defined by the [XML Specification 1.0].
NMOKENS	Whitespace-separated list of <i>NMOKEN</i> values.
Orientation	Enumeration that specifies named orthogonal two-dimensional orientations. See Table A-3, “Orientation Enumeration Values” on page 1155 for a list values.
Orientations	Whitespace separated list of <i>Orientation</i> enumeration values that specify named orthogonal two-dimensional orientations.
PDFPath	Whitespace-separated list of path operators as defined in PDF.
rectangle	Whitespace-separated list of four doubles representing a rectangle.
regExp	Regular expression as defined by [XMLSchema]
shape	Whitespace-separated list of three doubles representing a three-dimensional shape consisting of a width, height and length. Unless specified otherwise in the Attribute description, these three numbers are an X-dimension, a Y-dimension and a Z-dimension, respectively.
ShapeRange	Two <i>shape</i> values separated by a space character that defines a 3-dimensional box bounded by x1 y1 z1 x2 y2 z2.
ShapeRangeList	Whitespace-separated list of <i>shape</i> values or <i>ShapeRange</i> values. A shape value is represented by 2 shape values that are identical.
Source	xjdf or xml.
string	Character strings without tabs or line feeds. Corresponds to the standard XML normalized-String data type [XMLSchema].
text	Text data contained in an XML element (between start and end tag). A few Elements, such as <i>Comment</i> , have text.

Table 1-7: XJDF Data Types (Sheet 3 of 3)

Data Type	Description
text element	Element that contains text between start and end tags. e.g. <Comment>example text</Comment>.
TimeRange	Two <i>dateTime</i> values separated by a space character that defines the closed interval of the two. TimeRange corresponds semantically to the time interval (two time instants separated by a slash) defined in [ISO8601:2004].
TransferFunction	Whitespace separated list of an even number of doubles representing a set of XY coordinates of a transfer function.
URI	URI-reference. Represents a Uniform Resource Identifier (URI) Reference as defined in Section 4 of [RFC3986]. For the " <i>file</i> :" URL scheme, see [RFC3987]. URI includes Internationalized Resource Identifiers (IRI).
URL	URL-reference. Represents a Uniform Resource Locator (URL) Reference as defined in Section 4 of [RFC3986]. For the " <i>file</i> :" URL scheme, see [RFC3987]. URL includes usage of Internationalized Resource Identifiers (IRI).
WorkStyle	Specifies work styles of a press run. See Table A-6, "WorkStyle Enumeration Values" on page 1155 for a list values.
XPath	Represents an XPath expression of an XML node set (Attributes or Elements), boolean, double or string.[XPath]
XYPair	Whitespace-separated list of two doubles. Unless specified otherwise in the Attribute Description, these two doubles are an X-dimension and a Y-dimension, respectively.
XYPairRange	Two <i>XYPair</i> values separated by a space character that defines a rectangle bounded by x1 y1 x2 y2.
XYPairRangeList	Whitespace-separated list of <i>XYPairRange</i> values.
XYRelation	Defines the relationship between two ordered doubles. See Table A-7, "XYRelation Enumeration Values" on page 1156 for a list of NMOKEN values.

1.7 Units

XJDF specifies most values in default units. That means that an implementation SHALL use the defined default units and SHALL NOT use alternate units. All measurable quantities are stated in double precision. Processors SHOULD NOT specify a unit unless no default exists, such as when new Resources are defined. Then the units SHALL be based on metric units. Overriding the default units that are defined in this table is non-standard and MAY lead to undefined behavior. Any exceptions are specified in the appropriate descriptive tables.

The following table lists the units used in **XJDF**. The "Representation" column specifies the XML representation to indicate the units used in the following **XJDF** attributes:

- 1 The //ResourceSet/@Unit (see Table 8.1, "Resource" on page 517).
- 2 The //DeviceInfo/@CounterUnit (see Table 5-57, "DeviceInfo Element" on page 247):

Table 1-8: Units Used in XJDF (Sheet 1 of 2)

Measurement	Unit	Representation	Remarks
Length	point (1/72 inch)	pt	Used for all except microscopic lengths (see below)
	micron (μ)	um	Used for microscopic lengths — where used (instead of points) it will be explicitly stated in the definition of the item. See Media/@Thickness .
Volume	liter	l	—

Table 1-8: Units Used in XJDF (Sheet 2 of 2)

Measurement	Unit	Representation	Remarks
Weight	gram	g	—
Area	m ²	m2	Used for Media, (e.g., in wide format printing).
Resolution	dpi	dpi	The dots per inch (dpi) for print output and bitmap image (TIFF, BMP, etc.) file resolution.
Line Screen	lpi	lpi	The lines per inch (lpi) for conventionally screened halftone, screened grayscale and screened monotone bitmap images.
Screen Resolution	ppi	ppi	The pixels per inch (ppi) for screen display (e.g., soft-proof display and user interface display), scanner capture settings and digital camera settings.
Spot Resolution	spi	spi	For imaging Devices such as filmsetters, platesetters and proofers, the fundamental imaging unit (e.g., one “on” laser or imaging head imaged unit). Note: Many imaging Devices construct dots from multiple imaging spots, so dpi and spots per inch (spi) NEED NOT be equivalent.
Paper weight	g/m ²	gsm	Paper weight SHALL be provided in grams per square meter. See Appendix F, “North American and Japanese Media Weight Explained” on page 1189 for details of calculating non-gsm paper weights.
Speed	units/hour		Speed SHALL be specified in base units per hour. The base units that are used to represent speed SHALL be identical to the base unit.
Temperature	C° (Celsius)	C	degree centigrade
Angle	degrees°	degree	—
Countable Objects	1	count	Countable objects, such as Sheets, MAY be specified as “count”.

1.7.1 Counting in XJDF

Zero-based indices SHALL be used in **XJDF**. Thus the first index is 0, the second index is 1 etc. Note that this restriction applies to the **XJDF** representation only. Display of values, for instance in a user interface, is implementation defined.

Chapter 2 Overview of XJDF

Introduction

This chapter explains the basic aspects of **XJDF**. It outlines the terminology that is used and is recognized by the format, and the components of a workflow necessary to execute a printing Job using **XJDF**. Also provided is a brief discussion of **XJDF** Process structure and the role of messaging in an **XJDF** Job.

2.1 System Components

This section defines unique terminology used in this specification for the Job and workflow components of **XJDF**. Links to additional information is included for some terms.

The reader is assumed to have basic knowledge of XML syntax. Further information can be found at Table 1-1, “Basic References” on page 4.

2.1.1 Referencing External Data

External data is referenced from XJDF using standard URLs (Uniform Resource Locators) [RFC1738].

2.1.2 Identifying Sections of XJDF from External Sources

Certain aspects of a Job need to be identified from multiple XJDF or XJMF instances. Examples include XJDF/*@JobID*, XJDF/*@JobPartID* and *@ExternalID*. These entities do NOT have a data type of ID and systems that use XJDF SHOULD maintain them.

2.1.3 Identifying Sections of XJDF from within the Same XJDF

Certain aspects of a Job need to be identified from within a single XJDF instance. XML provides an ID - IDREF mechanism where an ID SHALL only be defined once within an XML instance and MAY be referenced multiple times by an IDREF or IDREFS from within the same XJDF instance.

All attributes in XJDF with a data type of ID SHALL be named ID. The reference types MAY have names other than IDREF. IDs and IDREFS are only valid within the scope of a single XJDF instance and NEED NOT be maintained when a new XJDF is generated.

2.1.4 Workflow Component Roles

The three components to create, modify, route, interpret and execute an **XJDF** Job are known as Controllers, Devices and Machines. The MIS or Management Information Systems is the top level controller in an XJDF workflow.

By defining these terms, this specification does not intend to dictate to manufacturers how to design, build or implement an **XJDF/XJMF** system. The intention is to name the component mechanisms needed for the interaction of actual components in a workflow during the course of an **XJDF** Job. In practice, it is very likely that individual system components will include a mixture of the capabilities described in the following sections.

2.1.4.1 Machines

A Machine is any part of the workflow system designed to execute a Process. Most often, this term refers to a piece of physical equipment, such as a press or a binder, but it can also refer to the software components used to run a particular Machine or perform a calculation. Computerized workstations, whether run through automated batch files or controlled by a human worker, are also considered Machines if they have no **XJDF** interface.

2.1.4.2 Devices

The most basic function of a **Device** is to execute the information specified and routed by a **Controller**. Devices SHALL be able to execute the instructions that are specified in **XJDF** and initiate **Machines** that can perform the physical execution. The communication between **Machines** and **Devices** is by definition proprietary and therefore not defined in this specification. **Devices** MAY, however, support **XJMF** messaging in order to interact dynamically with **Controllers**.

2.1.4.3 Queue

Whereas **Devices** process XJDF to produce a result, Queues provide a method of ordering, prioritizing and scheduling QueueEntries that represent XJDF processes. Every **Device** that is capable of accepting XJDF via XJMF messaging SHALL provide exactly one Queue. This specification makes no assumptions on implementation limitations of a Queue. Thus a device that can only process a single QueueEntry and cannot store any waiting QueueEntry still implements an albeit minimalistic Queue.

2.1.4.4 Controllers

Controllers route **XJDF** information to the appropriate **Devices**. The minimum requirement of a Controller is that it can initiate **Processes** on at least one Device, or at least one other Slave Controller that will then initiate **Processes** on a Device. In other words, a Controller is not a Controller if it has nothing to control. A pyramid-like hierarchy of Controllers can be built, with a Controller at the top of the pyramid controlling a series of lower-level Controllers at the bottom. The lowest-level Controllers in the pyramid, however, SHALL have **Device** capability. Therefore, Controllers SHALL be able to work in collaboration with other Controllers. In order to communicate with one another, and to communicate with **Devices**, Controllers SHALL support the **XJDF** file-exchange protocol and MAY support **XJMF**. Controllers can also determine Process planning and scheduling data, such as Process times and planned production amounts.

2.1.4.5 Management Information Systems—MIS

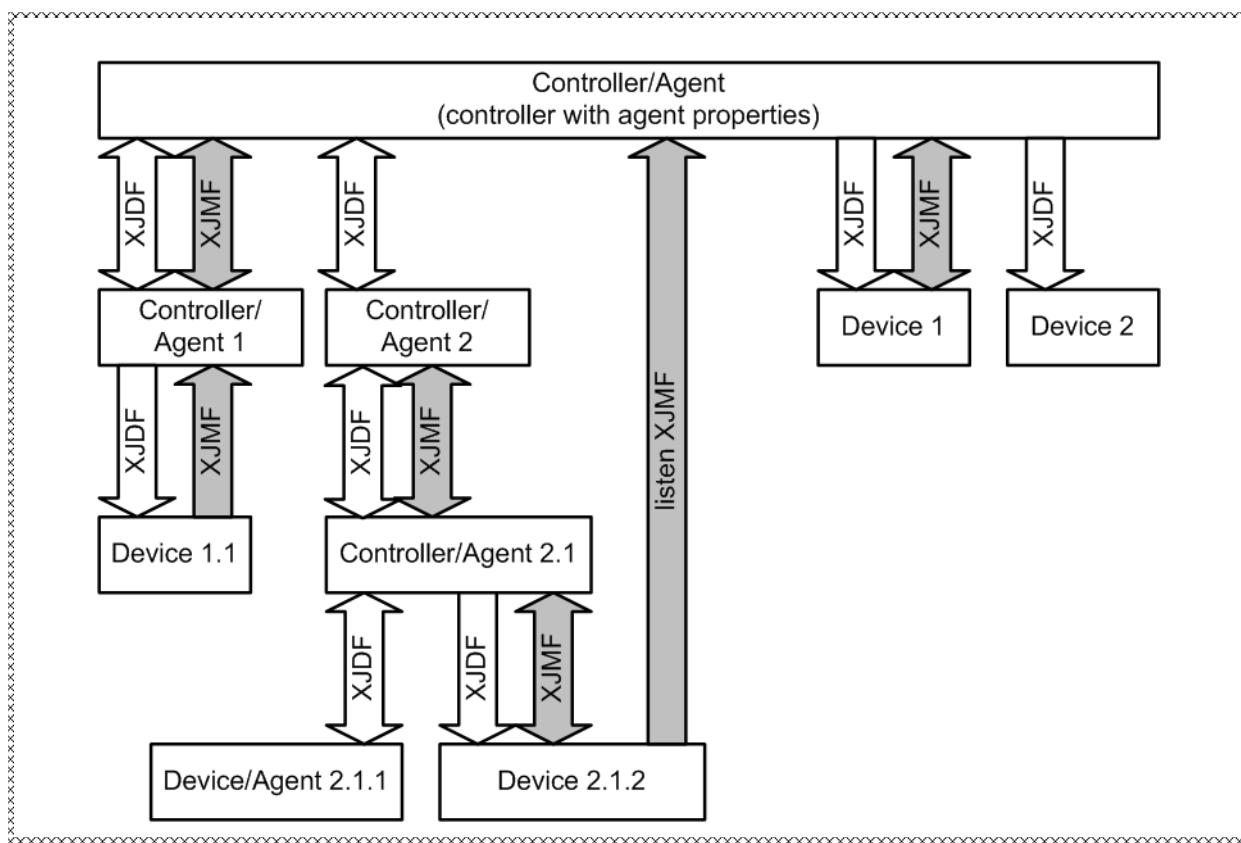
The highest level Controller in a workflow is known as a Management Information Systems or MIS. It is responsible for dictating and monitoring the execution of all of the diverse aspects of the workflow. This task is facilitated by access to production information, either in real time using **XJMF** messaging or post facto using the audit records within a returned **XJDF**.

To allow MIS to communicate effectively with the other workflow components, **XJDF** supplies what is essentially a messenger service, in the form of **XJMF**, to run between MIS and production. This format is equipped with a variety of Message types, ranging from simple, unidirectional notification to queries and even commands. System designers have a great deal of flexibility in terms of how they choose to use the messaging architecture, so that they can tailor the **Processes** to the capabilities of the existing workflow mechanism. The Figure 2-2 depicts how various communication threads can run between MIS and production.

XJDF also provides system components the ability to collect performance data, which can then be passed on to a Job-tracking system for use by the MIS system. These data can be derived from the Messages that the Controller receives or from the audit records in the Job. For more information on audits, see Section 3.16.4, “Audit”. Alternatively, the completed Job can be passed to the Job accounting system, which examines the audit records to determine the costs of the **Processes**.

2.1.4.6 System Interaction

An example of the interaction and hierarchical structure of the components considered in the preceding sections is shown in the following figure. Single arrows indicate unidirectional communication channels and double arrows indicate bidirectional communication.

Figure 2-1: Example of XJDF and XJMF workflow interactions

2.2 XJDF Workflow

XJDF does not dictate that a workflow be constructed in any pre-specified way. **XJDF** is equally as effective with a simple system using a single Controller and Device as it is with a completely automated industrial press workflow with integrated pre- and postpress operations.

An **XJDF** is defined in terms of inputs and outputs. The inputs of an **XJDF** consist of the materials it uses and the parameters that control it. For example, the inputs of an **XJDF** describing the Process parameters for imaging the cover of a brochure might include requirements for trapping, RIPing, and imposing the image. The output of our example **XJDF** might be a raster image.

A print job will typically require more than one process step to produce the final product. Each process step is completely defined by an **XJDF**. The interdependencies of the process steps MAY be specified in **XJDF** if the receiving device requires this information. Otherwise these interdependencies SHOULD remain opaque and be processed in a proprietary manner by the job Controller.

An **XJDF Job**, like any printing Job, is defined by the original intent for the end product. It is the task of the **MIS** to define the list of process that are most efficient for producing the desired product. The details of the process definition are implementation specific and not part of this specification.

2.2.1 Product Intent and Processes

XJDF describes a job from two points of view that are related but not identical.

- **Product Intent:** The customer or product designer will typically describe the desired final product without any knowledge of the manufacturing process. In **XJDF** this type of information is encoded in the **ProductList** and its child elements. Product intent is described in detail in Chapter 7, “Product Intents”

- **Process:** The devices that execute a processing step will typically receive processing instructions for that specific work as part of the manufacturing process for a product. In XJDF this type of information is encoded in **ResourceSet** and their child elements. Process resources are described in detail in Chapter 8, “Resources”.

Intent descriptions have been consciously limited to details of the more common products. In order to reduce duplication of resources and keep intent definitions simple, some features that are typically required to describe products that are used in business to business workflows such as packaging have not been included in Intent descriptions. These specialized products SHOULD be described by adding process resources that describe the desired features.

Controllers such as MIS SHOULD evaluate product intent and provide all processing instructions for devices as **ResourceSet** elements. Devices NEED NOT evaluate product intent to infer processing instructions. Product intent is provided to Devices in order to provide operators with an overview of the context of the process step within one or more customer jobs.

XJDF is an interface format that specifies the exchange of information amongst Controllers and Devices in a workflow.

2.3 Role of Messaging in XJDF

Whereas **XJDF** will typically be submitted to a Device and only be returned after the process has been executed, **XJMF** provides methods to dynamically synchronize and manipulate Controllers and Devices. For more details on XJMF, see Chapter 5, “JMF XJMF Messaging with the Job Messaging Format” and Chapter 11, “Building a System Around JDF XJDF”.

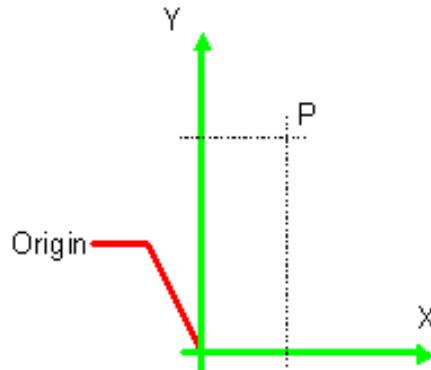
2.4 Coordinate Systems in XJDF

This chapter explains how coordinate systems are defined and used in **XJDF**. It also shows how the matrices are used to specify a certain transformation and how these matrices can be used to transform coordinates from one coordinate system to another coordinate system. In addition, it clarifies the meaning of terms like “*Top*” or “*Left*”.

2.4.1 Introduction

During the production of a printed product it often happens that one object is placed onto another object. During imposition, for example, single pages and marks (like cut, fold or register marks) are placed on a Sheet surface. Later, at image setting, a bitmap containing one separation of a Sheet surface is imposed on a piece of film. In a following step, the film is copied to a printing plate which then is mounted on a press. In postpress, the printed Sheets are gathered on a pile. The objects involved in all these operations have a certain orientation and size when they are put together. In addition, one has to know *where* to place one object on the other.

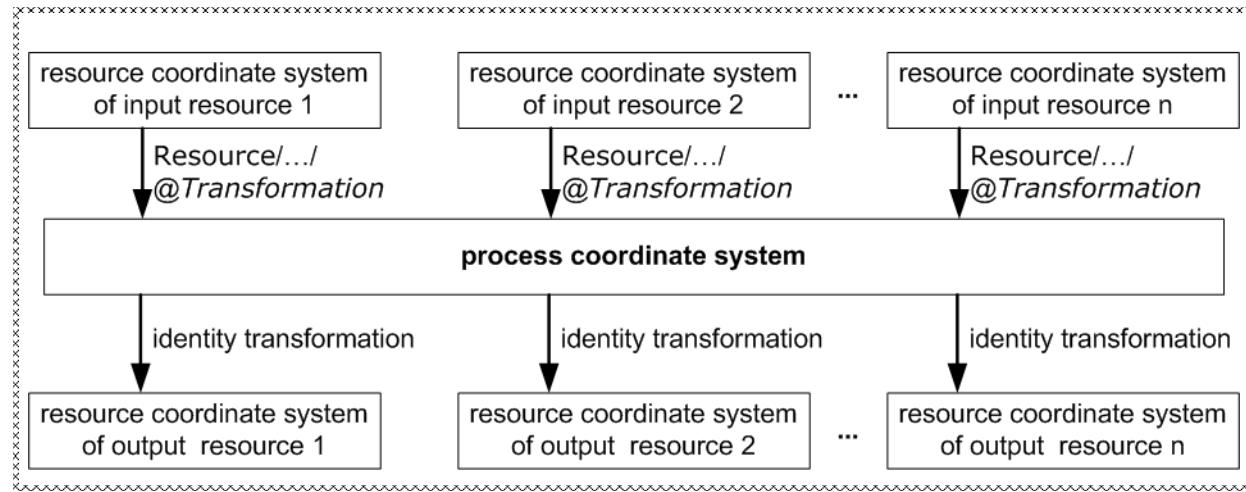
The position of an object (e.g., a cut mark) on a plane can be specified by a two-dimensional coordinate. Every digital or physical **Resource** has its own coordinate system. The origin of each coordinate system is located in the lower left corner (i.e., the X coordinate increases from left to the right, and the Y coordinate increases from bottom to top).

Figure 2-2: Standard coordinate system

Each page contained in a PDL file has its own coordinate system. In the same way a piece of film or a Sheet of paper has a coordinate system. Within **XJDF** each of these coordinate systems is called *Resource coordinate system*.

If a Process has more than one Input Resource with a coordinate system, it is necessary to define the relationship between these input coordinate systems. Therefore, a *process coordinate system* is defined for each Process. **XJDF** tickets are written assuming an idealized Device that is defined in the process coordinate system for each Process that the Device implements. A real Device SHALL map the idealized process coordinate system to its own device coordinate system.

The coordinate systems of the Input Resources are mapped to the process coordinate system. Each of those mappings is defined by a transformation matrix, which specifies how a coordinate (or position) of the input coordinate system is transformed into a coordinate of the target coordinate system. (See Section 2.5.6, “Homogeneous Coordinates” for mathematical background information.) In the same way, the mapping from the process coordinate system to the coordinate systems of the Output Resources is defined. The process coordinate system is also used to define the meaning of terms like “*Top*” or “*Left*”, which are used as values for parameters in some Processes.

Figure 2-3: Relation between Resource and process coordinate systems

It is important that no implicit transformations (such as rotations) are assumed if the dimensions of the Input Resources of a Process do not match each other. Instead every transformation (e.g., a rotation) SHALL be specified explicitly by using the *@Orientation* or *@Transformation* Attribute of the corresponding **Resource/AmountPool/PartAmount**. The same applies also to other areas in **XJDF** (e.g., the *Interpreting* Process). A FitPolicy Element MAY define a policy for implied transformations.

2.4.1.1 Source Coordinate Systems

The source coordinate system of a referenced object is defined by the lower left of the object. X values are increasing to the right, Y values are increasing towards the top. In case of PDF the lower left of the MediaBox defines the lower left of the source coordinate system.

Note: some object coordinate systems have optional tags to indicate internal transformations. These internal transformations SHALL be applied prior to defining the source coordinate system; for instance:

- PDF: the rotation defined by the Rotate key SHALL be applied. The lower left of the MediaBox of the rotated PDF defines the lower left of the PDF source coordinate system.
- TIFF: the orientation defined by the Orientation tag SHALL be applied. The lower left of the rotated TIFF defines the lower left of the TIFF source coordinate system.

2.4.2 Coordinates and Transformations

Table 2-1: Data types for specifying coordinates and transformation

Data Type	Example
XYPair	"612 792"
double	"20.7"
rectangle	"0 0 595 843" (Order of elements is "lower-left x, lower-left y, upper-right x, upper-right y" or "left, bottom, right, top".)
Matrix	"1 0 0 1 30.0 235.3" The ordering of elements is defined in Section 2.5.6, "Homogeneous Coordinates".
Orientation	"Rotate180" or "Flip90"

Coordinates and transformations are used throughout **XJDF**, to include:

Intent Resources, such as:

- **LayoutIntent**: specifies size of finished product
- **MediaIntent**: specifies size of media
- **InsertingIntent**: specifies rotation and offset of inserts

Process Resources, such as:

- **Component**: specifies coordinate system
- **CutBlock**: specifies cut block coordinate system
- **FoldingParams**: specifies folding operations

2.4.3 Coordinate Systems of Resources and Processes

Each physical Input Resource (e.g., **Component**) of a Process has, by default, its own coordinate system, which is called the "resource coordinate system." The coordinate system also implies a specific orientation of that **Resource**. On the other hand there is a coordinate system that is used to define various Process-specific parameters. This coordinate system is called a target or process coordinate system.

It is often necessary to change the orientation of an Input Resource before executing the operation. This can be done by specifying a transformation matrix. It is stored in the **@Orientation** or **@Transformation** Attribute of the **Resource**. This provides the ability to specify different matrices for the individual Resources of a Process. For details on **Resource**, **AmountPool** and **PartAmount** Elements, see Section 8.1, "Resource" on page 517.

2.4.3.1 Coordinate Systems of Combined Processes

Processes MAY specify multiple individual Processes' and thus also the Processes respective coordinate systems. The process coordinate systems are not modified by the fact that the Processes are part of a Combined Process, they are identical to the process coordinate systems of the Processes, were they defined in a linked chain of individual Processes. The coordinate systems of an exchange Resource can be modified by defining it as a pipe by specifying `Dependent@PipeID` and `Dependent@PipeProtocol = "Internal"` (See Section 4.4.3, “Overlapping Processing Using Pipes” on page 191) and linking it to the Combined Process with both an input and outputResource. The Input Resource defines the coordinate transformation using the standard `@Transformation` or `@Orientation` Attributes.

2.4.3.2 Coordinate System Transformations

The following table shows some matrices that can be used to change the orientation of a **Resource**. Most of the transformations require the X- (**w**) and the Y-dimension (**h**) of the **Component** as specified in the `@Dimensions` Attribute. If these are unknown, it is still possible to define a general orientation in the `@Orientation` Attribute of the **Resource**. The naming of the Attribute reflects the state of the Resource and not necessarily the order of applied transformations. Thus `"Rotate90"` and `"Flip90"` specify that the original Y axis as represented by the spine is on top. In the case of `Flip90`, the **Component** is additionally flipped front to back.

Table 2-2: Matrices and Orientation values for describing the orientation of a Component (Sheet 1 of 2)

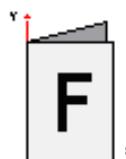
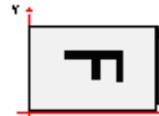
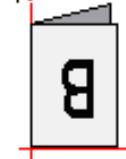
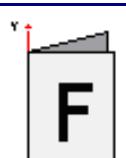
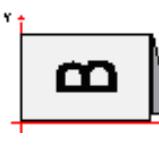
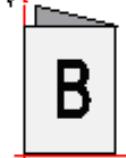
Orientation Value	Source Coordinate System	Transformation Matrix According Action	Target Coordinate System
Rotate0		$1 \ 0 \ 0 \ 1 \ 0 \ 0$ No Action	
Rotate90		$0 \ 1 \ -1 \ 0 \ h \ 0$ 90° Counterclockwise Rotation	
Rotate180		$-1 \ 0 \ 0 \ -1 \ w \ h$ 180° Rotation	
Rotate270		$0 \ -1 \ 1 \ 0 \ 0 \ w$ 270° Counterclockwise Rotation	
Flip0		$1 \ 0 \ 0 \ -1 \ 0 \ h$ Flip around X	
Flip90		$0 \ -1 \ -1 \ 0 \ h \ w$ 90° Counterclockwise Rotation + Flip around X	
Flip180		$-1 \ 0 \ 0 \ 1 \ w \ 0$ 180° Rotation + Flip around X	

Table 2-2: Matrices and Orientation values for describing the orientation of a Component (Sheet 2 of 2)

Orientation Value	Source Coordinate System	Transformation Matrix According Action	Target Coordinate System
<i>Flip270</i>		$\begin{matrix} 0 & 1 & 1 & 0 & 0 & 0 \end{matrix}$ 270° Counterclockwise Rotation + Flip around X	

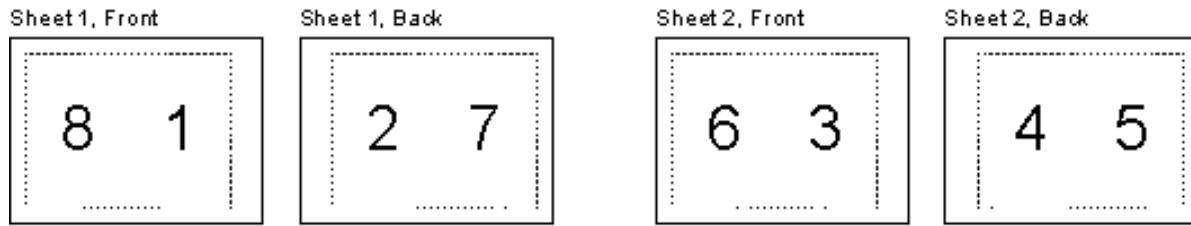
2.4.4 Product Example: Simple Brochure

To illustrate the use of coordinate systems in XJDF, a simple saddle stitched brochure with eight pages is used as an example in Table 2-5. The brochure is printed on two Sheets with front and back. The two Sheets are then folded, collected on a saddle, and saddle stitched. Finally the brochure is cut with a three-side trimmer.

Table 2-3: XJDF Processes used for the production of the simple brochure

Input Resources	Process	Output Resources
Layout RunList (Document) RunList (Marks)	Imposition	RunList
RunList	Interpreting	RunList (of interpreted PDL data)
RunList (of interpreted PDL data) Media RenderingParams	Rendering	RunList (of rasterized byte maps)
RunList (of rasterized byte maps)	Screening	RunList (of bit maps)
ImageSetterParams Media (of film) RunList (of bit maps)	ImageSetting (to film)	ExposedMedia (of film)
ExposedMedia (of plate) ConventionalPrintingParams	ConventionalPrinting	Component
FoldingParams Component	Folding	Component
CollectingParams Component	Collecting	Component
StitchingParams Component	Stitching	Component
TrimmingParams Component	Trimming	Component

At imposition, the layout describes a Signature with two Sheets, each having a front and a back surface. On each surface, two content objects (i.e., pages, are placed).

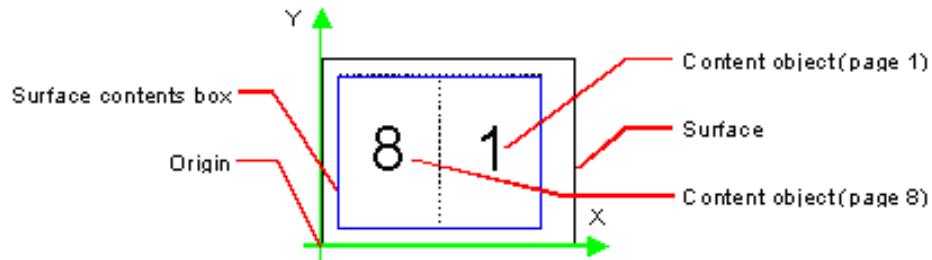
Figure 2-4: Layout of simple saddle stitched brochure (product example)

Each surface has its own coordinate system, in which a surface contents box is defined. This coordinate system is also referred to as the **Layout** coordinate system because the Signature, Sheet and surface elements are defined within the hierarchy of the **Layout** Resource by means of Partitioning. The content objects are placed by specifying the CTM Attribute relative to the surface contents box. If the position of an object within a page is given in the page coordinate system, this coordinate can be transformed into a position within the surface coordinate system:

Figure 2-5: Equation for Surface Coordinate System Transformations

$$P_{\text{Surface}} = P_{\text{Page}} \times \text{CTM}_{\text{Page}} + [\text{SurfaceContentsBox}_{x\text{lowerleft}} \text{ SurfaceContentsBox}_{y\text{lowerleft}} \ 0]$$

Please note, that the width and height of the surface NEED NOT be known at this point.

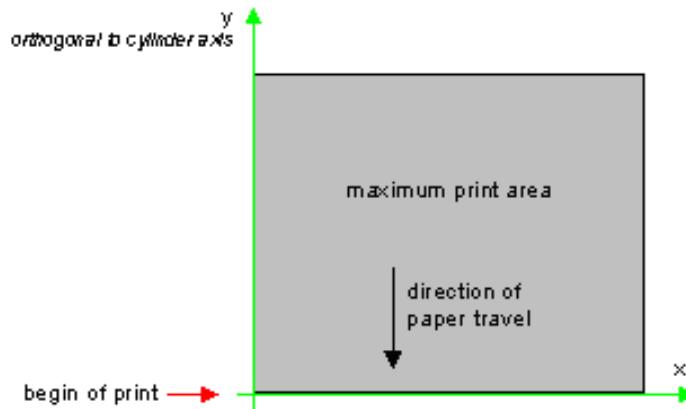
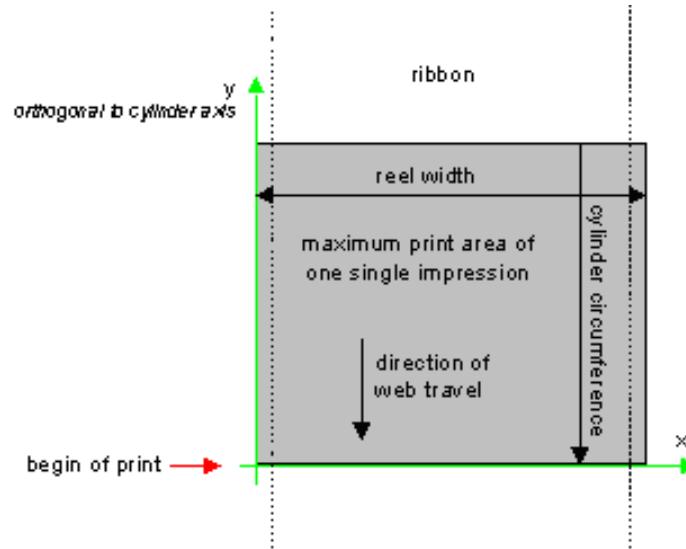
Figure 2-6: Surface coordinate system

The Sheet coordinate system is identical with the coordinate system of the front surface. This means that no transformation is needed to convert a coordinate from one system to the other. Instead, the coordinates are valid (and equal) in both coordinate systems. The relation between the coordinate system of the front and the back surfaces depends on the value of the **Layout/@LockOrigins** Attribute. The Sheet coordinate system is also identical with the Signature coordinate system, which in turn is identical with the coordinate system of the **Imposition** Process.

The Output Resource of the **Imposition** Process is a run list. Each Element of the run list has its own coordinate system, which is identical with the corresponding Signature coordinate system. The interpretation, rendering and screening Processes do not affect the coordinate systems. This means that the coordinate systems of all these Processes are identical.

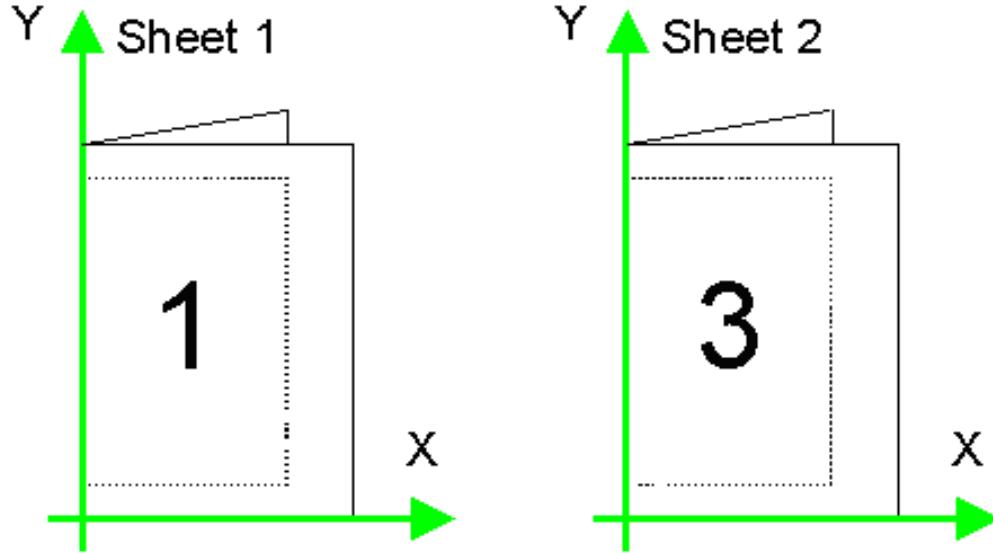
At the image setting Process, the digital data is set onto film. The process coordinate system is defined by the media Input Resource. The width and height of the media are defined in the **Media/@Dimension** Attribute. The position of the Signatures (as defined by the run list Input Resource) on the film is defined by the **ImageSetterParams/@CenterAcross** Attribute.

The coordinate system of the conventional and digital printing Processes is called *press coordinate system*. It is defined by the press: the X-axis is parallel to the press cylinder, and the Y-axis is going along the paper travel. Y = 0 is at begin of print, X = 0 is at the left edge of the maximum print area. The Front side of the press Sheet faces up towards the positive Z-axis. The relationship between the layout coordinate system and the press coordinate system is defined by the **@CTM** Attributes of the corresponding **TransferCurve** Elements.

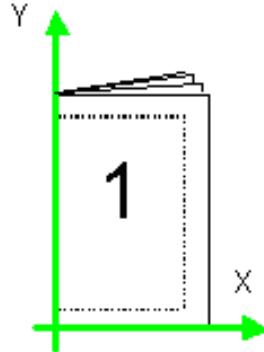
Figure 2-7: Press coordinate system used for Sheet-Fed Printing**Figure 2-8: Press coordinate system used for Web Printing**

The output of the printing Process (e.g., a pile of printed Sheets) is described as a **Component** Resource in XJDF. The coordinate system of the printed Sheets is defined by the transformation given in the **TransferCurve/@CTM** Attribute (where **Part/@TransferCurveName = "Paper"**).

Each of the two Sheets is folded in a separate folding Process. In this example, the orientation of the Sheets is not changed before folding. This can be specified by setting the **@Orientation** Attribute of the Input Resource to "*Rotate0*" or by setting the **@Transformation** Attribute to "*1 0 0 1 0 0*". The folding Process changes the coordinate system. In this example the origin of the coordinate system is moved from the lower left corner of the flat Sheet (input) to the lower left corner of the folded Sheet (output) (i.e., it is moved to the right by half of the Sheet width).

Figure 2-9: Coordinate systems after Folding (product example)

The two folded Sheets are now collected. In this example, the orientation of the folded Sheets is not changed before collecting. This can be specified by setting the *@Orientation* Attribute of the Input Resource to "Rotate0" or by setting the *@Transformation* Attribute to "1 0 0 1 0 0". The collecting Process does not change the coordinate system.

Figure 2-10: Coordinate systems after Collecting (product example)

The two collected and folded Sheets are now trimmed to the final size of the simple brochure. In this example, the orientation of the collected and folded Sheets is not changed before trimming. This can be specified by setting the *@Orientation* Attribute of the Input Resource to "Rotate0" or by setting the *@Transformation* Attribute to "1 0 0 1 0 0". The trimming Process changes the coordinate system: the origin is moved to the lower left corner of the trimmed product.

In looking at the whole production Process, a series of coordinate systems is being involved. The relationship between the separate coordinate systems is specified by transformation matrices. This allows transformation of a coordinate from one coordinate system to another coordinate system. As an example, note the position of the title on page 1 of the product example in Figure 2-15. By applying the first transformation, this position can be converted into a position of the surface (or layout) coordinate system. This position can then be converted into the paper coordinate system by applying (in this order) the *"Film"*, *"Plate"*, *"Press"* and *"Paper"* transformations stored in the **TransferCurve** Elements.

From now on in the workflow, every Process is using components as input and Output Resources. The **Resource** of each input and output component contains a *@Transformation* Attribute or an *@Orientation*

Attribute. The *@Transformation* Attribute SHALL be used if the width and the height of the component are known or a non-orthogonal rotation is needed. Otherwise the *@Orientation* Attribute SHOULD be used to specify a change of the orientation (e.g., an orthogonal rotation).

Since the folding Process changes the coordinate system depending on the fold type, the transformations specified in the **Resource** Elements are not sufficient to transform a position given in the paper coordinate system to a position in the coordinate system of the folded Sheets (i.e., the resource coordinate system of the output component of the folding Process). An additional transformation depending on the fold type and details of the individual folds has to be applied. The corresponding transformation matrix is not explicitly specified in the **XJDF** file.

The collecting Process does not change the coordinate system. Therefore, only the transformations specified in the **Resource** Elements of the Input and Output Resources (i.e., components have to be applied).

The trimming Process again changes the coordinate system depending on the trimming parameters. Therefore, a transformation depending on the trimming parameters has to be applied in addition to the transformations specified in the **Resource** Elements. The matrix for the additional transformation (depending on the trimming parameters) is not explicitly specified in the **XJDF** file.

After having applied all transformations mentioned above, the resulting coordinate specifies the position of the title in the coordinate system of the final product.

Figure 2-11: Examples of Transformations and Coordinate Systems in XJDF.



2.4.5 General Rules

The following rules summarize the use of coordinate systems in **XJDF**.

- Every individual piece of material (film, plate, paper) has a *resource coordinate system*.
- Every Process has a *process coordinate system*.
- Terms like *top*, *left*, etc., are used with respect to the *process coordinate system* in which they are used and are independent of orientation (i.e., *landscape* or *portrait*), and the human reading direction.
- The coordinate system of each input component is mapped to the process coordinate system.
- The coordinate system might change during processing (e.g., in **Folding**).
- The description of a product in **XJDF** is independent of particular Machines used to produce this product. When creating setup information for an individual Machine, it might be necessary to compensate for certain Machine characteristics. At printing, for example, it might be necessary to rotate a landscape Job because the printing width of the press is not large enough to run the Job without rotation.

2.4.6 Homogeneous Coordinates

A convenient way to calculate coordinate transformations in a two-dimensional space is by using so-called homogeneous coordinates. With this concept, a two-dimensional coordinate $P=(x,y)$ is expressed in vector form as $[x \ y \ 1]$. The third element "1" is added to allow the vector being multiplied with a transformation matrix describing scaling, rotation, and translation in one shot. Although this only requires a 2×3 matrix (e.g., as it is used in PostScript) in practice 3×3 matrices are much more common, because they can be concatenated very easily. Thus, the third column is set to " $0 \ 0 \ 1$ ".

$$\text{Trf} = \begin{bmatrix} a & b & 0 \\ c & d & 0 \\ e & f & 1 \end{bmatrix} \quad \text{would in XJDF be written as } "a \ b \ c \ d \ e \ f"$$

Some often used transformation matrices are

$$\text{Trf} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{identity transformation}$$

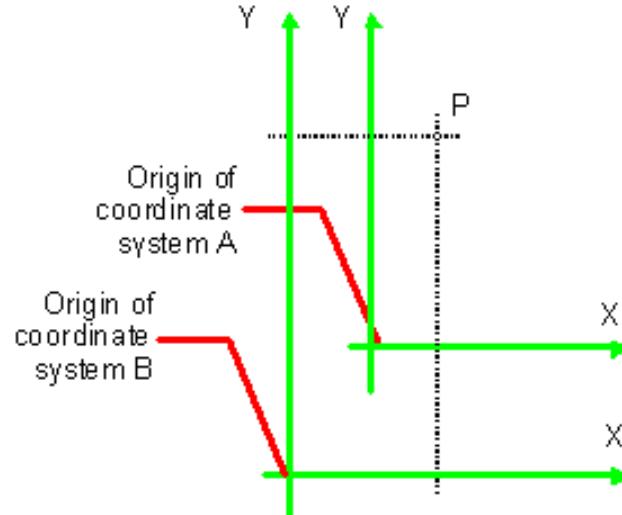
$$\text{Trf} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ dx & dy & 1 \end{bmatrix} \quad \text{translation by } dx, dy$$

$$\text{Trf} = \begin{bmatrix} \cos \varphi & \sin \varphi & 0 \\ -\sin \varphi & \cos \varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{rotation by } \varphi \text{ degrees counter-clockwise}$$

Transforming a point

In this example, the position P given in the coordinate system A is transformed to a position of coordinate system B. The relationship between the two coordinate systems is given by the transformation matrix Trf

Figure 2-12: Transforming a point (example)



$$P_A = (30, 100)$$

$$P_A = [30 \ 100 \ 1]$$

$$P_B = P_A \times Trf$$

in **XJDF**, *Trf* is written as "1 0 0 1 40 60"

$$P_B = [30 \ 100 \ 1] \times \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 40 & 60 & 1 \end{bmatrix}$$

$$P_B = (70, 160)$$

$$P_B = [70 \ 160 \ 1]$$

Chapter 3 Structure of XJDF

Introduction

A single XJDF Node describes the information about a Job or process step that is transferred from a Controller to a Device. The scope of the exchanged information varies depending on the nature of the recipient Device. An XJDF Node that is targeted at an individual Device will typically contain only the details that are required by that Device, along with some optional information about the final product. Multiple work steps belonging to one job that need to be submitted from a Controller to a workflow system that controls multiple Devices SHALL be submitted as multiple XJDF nodes, which MAY be submitted as one or more transactions. See chapter #ref building a system for details of packaging and referencing of the individual nodes.

XJDF

This section provides details of the XJDF element and its direct child elements.

3.1 Modification Notes

This chapter has the most significant modification compared to JDF 1.x. The syntax of Resource partitioning has been completely revised, and the concept of abstract Element types has been replaced by explicit Element definitions. Details are provided in the relevant subsections.

3.2 XJDF Node

3.2.1 Modification Notes

Syntax and process model for XJDF have been revised. The concept of test running has been removed. Section 5.5.3, “KnownDevices”) SHOULD be used to query the abilities of a Device.

XJDF Nodes are no longer nested. There is exactly one XJDF Node per XJDF ticket. Multiple Nodes^{%%%} with different @JobPartId MAY be sent to a controller to specify multiple individual tasks.

Product descriptions are now specified as a ProductList Subelement in the XJDF Node. This allows informative specification of one or more products for any process node without requiring the process to be a descendent of the respective product.

All XJDF Nodes are essentially Gray Boxes that can be processed by a Device. There are no Gray Box expansion requirements in case a Gray Box is processed by lower level Devices.

Resources have been split into two classes. Product Intent Elements are specified within their respective Product elements. All other Resource classes from JDF 1.x have been combined into the ResourceSet/Resource group. All generic Attributes and Elements SHALL be specified in the Product Intent or Resource element, whereas specific Attributes and Elements SHALL be specified in a corresponding Product Intent or Specific Resourcer as specified in Chapter 7, “Product Intents” or Chapter 8, “Resources”, respectively. Partitioning has been limited syntactically to exactly one level. Zero or more Part Elements specify the part usage, and each Part Element MAY still contain multiple partition attributes. Multiple Part Elements replace the Identical Element.

3.2.2 XJDF Node

The top-level Element of a XJDF instance SHALL be an XJDF Element. See Table 3-1 below for details. XJDF elements MAY be embedded within other XML documents.

XJDF/@*Types* defines whether an XJDF specifies an end product or a list of processes that SHALL be executed. XJDF Nodes that are created by print buyers typically describe only the desired Product rather than manufacturing process details. XJDF that describe finished products SHALL have a value of XJDF/@*Types* that contains "Product". If additional process information that is not defined in the ProductList is required, this information SHOULD be provided in ResourceSet Elements.

ProductList MAY be provided in a process XJDF for informational purposes. If the ResourceSet elements contain all required processing instructions then the value of XJDF/@*Types* SHALL NOT contain "Product".

Table 3-1: XJDF Node (Sheet 1 of 3)

Name	Data Type	Description
<i>Category</i> ?	NMTOKEN	<p>@<i>Category</i> specifies the named category of this XJDF. Controllers SHOULD specify @<i>Category</i> for processes that have many optional values in @<i>Types</i>. This allows Processors to identify the general purpose of a Node without parsing the @<i>Types</i> field. For instance, a RIP for final output and RIP for proof Process have identical @<i>Types</i> Attribute Values, but have @<i>Category</i> = "ProofRIPing" or @<i>Category</i> = "RIPing", respectively.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Binding</i> – Binding of a bound product. <i>Cutting</i> – Specifies cutting of a Component. <i>DigitalPrinting</i> – A RIP and print run on a digital printer that produces final output. <i>FinalImaging</i> – A RIP and image that produces final output that is ready for further processing (e.g., film or plates). <i>FinalRIPing</i> – RIP Process for generating final output. <i>Folding</i> – Folding of a product. <i>Newsprinting</i> – A press run on a newsprinting Web Press. <i>PostPress</i> – General postpress. Includes "Folding" and "Binding". <i>PrePress</i> – General prepress. <i>Printing</i> – A press run that produces final output. <i>ProofImaging</i> – A RIP that produces proof output. <i>ProofRIPing</i> – RIP Process for generating a proof. The Processes are identical to those in specified for "FinalRIPing". <i>RIPing</i> – General RIP Gray Box. For details, see Section 6.4.20.1, "RIPing". %%% <i>WebPrinting</i> – A press run on a Web Press can produce one or more components as output at the same time. A Web Printing press might be equipped with Prepress and Postpress equipment. <i>WebToPrint</i> – A Product description that describes a product order in a web shop. <p>Note: the value MAY also be the name of a Gray Box defined by an ICS document or XJDF spec. See the ICS documents for the exact names.</p>
<i>CommentURL</i> ?	URL	URL SHALL refer to an external, human-readable description of this XJDF.

Table 3-1: XJDF Node (Sheet 2 of 3)

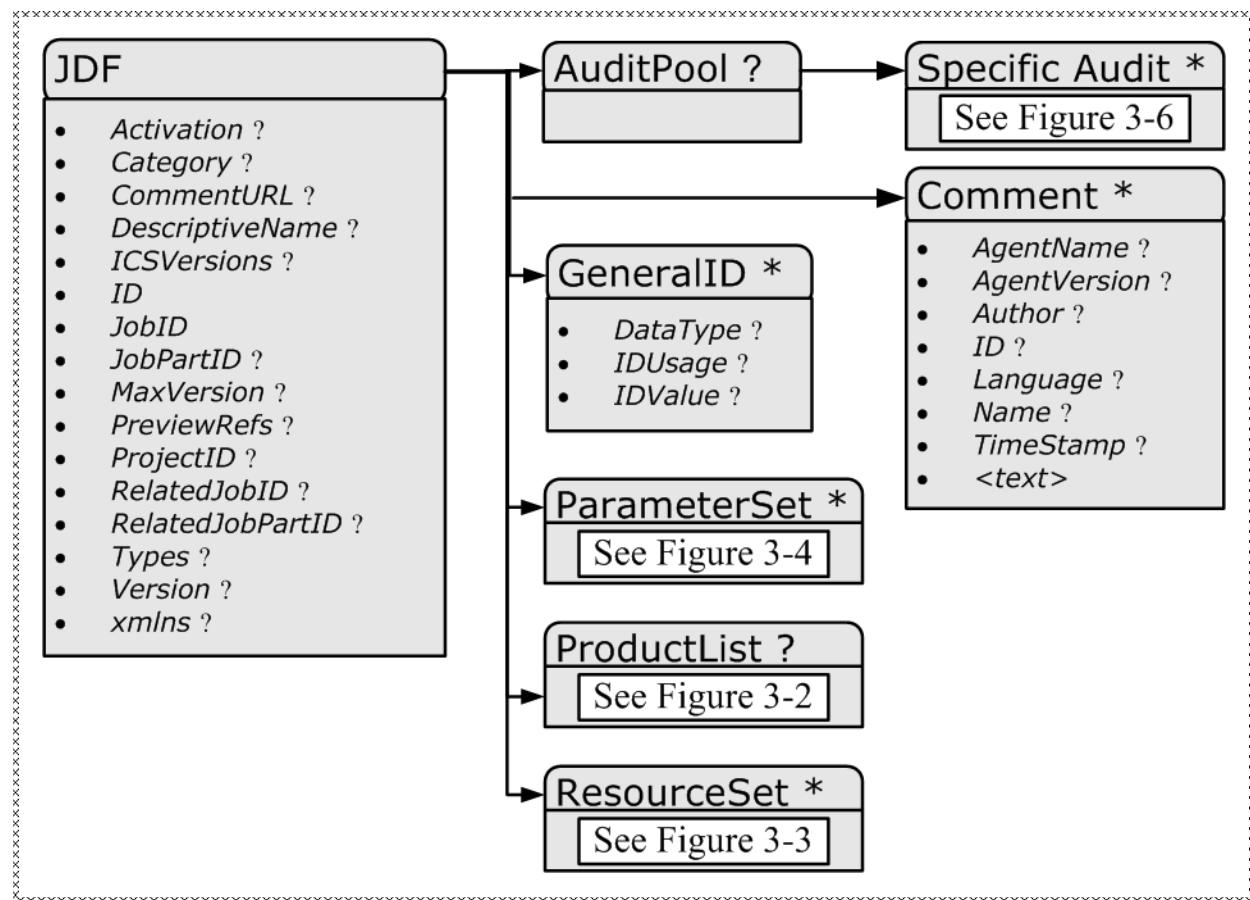
Name	Data Type	Description
<i>DescriptiveName</i> ?	string	Human-readable descriptive name of this XJDF. <i>@DescriptiveName</i> SHOULD be provided for communication from applications to humans in order to reference the XJDF.
<i>ICSVersions</i> ?	NMTOKENS	<p><i>@ICSVersions</i> SHALL list all CIP4 Interoperability Conformance Specification (ICS) Versions that this XJDF complies with.</p> <p>Value format is: <ICSName>_L<ICSLevel>-<ICSVersion>.</p> <p>Example: MISPRE_L1-2.0 for the MIS to Prepress ICS. See Section 11.9, “Interoperability Conformance Specifications” on page 1140 %/%/% for more information on ICS documents.</p>
<i>JobID</i>	NMTOKEN	Job identification used by the application that created the XJDF Job. Typically, a Job is identified by the internal order number of the MIS system that created the Job.
<i>JobPartID</i> ?	NMTOKEN	<i>@JobPartID</i> SHALL identify one or more worksteps of the same type that can be described as one XJDF . <i>@JobPartID</i> is internal to the MIS system that created the XJDF .
<i>ProjectID</i> ?	NMTOKEN	Identification of the project context that this XJDF belongs to. Used by the MIS to group a set of XJDF Jobs.
<i>RelatedJobID</i> ?	NMTOKEN	Job identification of a related Job. Used to identify the <i>@JobID</i> of a previous run of this Job or Job with very similar settings. It MAY be used to retrieve additional Job and Device specific settings from a data store.
<i>RelatedJobPartID</i> ?	NMTOKEN	Job identification of a related Job Part. Used to identify the <i>@JobPartID</i> of a previous run of this Job or Job with very similar settings. It MAY be used to retrieve additional Job and Device specific settings from a data store.
<i>Types</i>	NMTOKENS	A single or a list of process names that are specified within this XJDF Document. For details on using Process Nodes, see Section 3.3.3, “Process Nodes”. A value of <i>@Types</i> that contains “ <i>Product</i> ” specifies that the products that are described shall be produced without complete knowledge of the production workflow. Additional process specifics MAY be supplied in a ticket that contains “ <i>Product</i> ”. Values include those from: Chapter 6, “Processes” on page 355.
<i>Version</i>	JDFJMVersion	The version of the XJDF Node. The value SHALL be “2.0”.
<i>xmlns</i>	URI	XJDF supports use of XML namespaces. For details on using namespaces in XML, see [XMLNS]. For versions 2.0 of XJDF, <i>@xmlns</i> = “ http://www.CIP4.org/JDFSchema_2_0 ”.
<i>AuditPool</i> ?	element	List of Elements that contains all relevant audit information. Audit Elements are intended to serve the requirements of MIS for evaluation and post calculation. See Section 3.7, “AuditPool and Audit”.
<i>Comment</i> *	element	Any human-readable text. The Comment Element is different from an XML comment <!-- XML Comment -->. The XJDF comment is meant for display in a user interface whereas the XML comment is used to add developers comments to the underlying XML.
<i>GeneralID</i> *	element	Additional identifiers related to the XJDF.
<i>ProductList</i> ?	element	Bill of materials - description of the product, or products that this XJDF produces.

Table 3-1: XJDF Node (Sheet 3 of 3)

Name	Data Type	Description
ResourceSet *	element	Container Elements for Resource Elements.

3.2.3 Structure Diagram of XJDF Node

Figure 3-1 shows the structure of the XJDF Node. Arrows point to child elements.

Figure 3-1: XJDF Node – a Diagram of its Structure

3.3 Common Elements

3.3.1 Element: Comment

Table 3-2: Comment Element (Sheet 1 of 2)

Name	Data Type	Description
AgentName ?	string	The name of the application that created the Comment . Both the company name and the product name MAY appear, and SHOULD be consistent between versions of the application.
AgentVersion ?	string	The version of the application that created the Comment . The format of the version string MAY vary from one application to another, but SHOULD be consistent for an individual application.

Table 3-2: Comment Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Author</i> ?	string	Text that identifies the person who created the Comment . When the Comment is created by a person with a known Contact / @UserID , this @Author SHOULD contain the value of Contact / @UserID .
<i>ExternalID</i> ?	NMTOKEN	Identification that is used to reference the Comment .
<i>Language</i> ?	language	Human readable language of the Comment .
<i>Name</i> ?	NMTOKEN	<p>A name that defines the usage of a comment. For example, it could determine whether two comments are intended to fill two distinct fields of a user interface.</p> <p>Values include:</p> <p><i>Description</i> – Human readable description, which is REQUIRED if the Comment Element is REQUIRED in a given context, as is the case in the Notification %%%Element (see Table 3-44, “Notification Audit Element” on page 160).%%%</p> <p><i>DeviceText</i> – Human readable description created by the Device that provides details beyond the value of @StatusDetails.</p> <p><i>Instruction</i> – Message to the operator that contains information regarding the processing of the Job.</p> <p><i>JobDescription</i> – Description of the Job. A Comment Element that contains @Name = "JobDescription" SHALL be specified only in a JDF Node or CustomerInfoThe CustomerInfo Resource contains information about the customer who orders the Job. Resource. See also CustomerInfoThe CustomerInfo Resource contains information about the customer who orders the Job./@CustomerJobName in Section , “CustomerInfoThe CustomerInfo Resource contains information about the customer who orders the Job.”.</p> <p><i>OperatorText</i> – Message from the operator that contains information regarding the processing of the Job.</p> <p><i>Orientation</i> – Description of the orientation of a Resource. The “Orientation” value SHOULD only be specified in the context of a Resource.</p> <p><i>UserText</i> – Message to a user that contains information regarding the processing of the Job</p>
<i>PersonalID</i> ?	NMTOKEN	Identifier of the employee that entered the comment.
<i>TimeStamp</i> ?	dateTime	Describes the date and time when the Comment was created.
	text	Body of the comment. Note that whitespace is preserved only as generic whitespace in XML. Applications that display comments to the user SHOULD maintain whitespace. Thus carriage returns, line feeds or indentation MAY be lost.

3.3.2 Element: GeneralID

GeneralID describes a generic identifier. The name or usage of the identifier is specified in **GeneralID/@IDUsage** and the specific value of the variable is specified in **GeneralID/@IDValue**. The data type is specified in **GeneralID/@DataType**.

Although **GeneralID** could technically be used to describe arbitrary proprietary data, this is strongly discouraged as it is non interoperable. Proprietary extensions SHOULD be avoided if possible, or if absolutely required, they MAY be implemented in proprietary namespaces.

Table 3-3: GeneralID Element

Name	Data Type	Description
<i>DataType</i> ?	enumeration	<p>Data type of the variable.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>string</i> <i>integer</i> <i>double</i> <i>NMOKEN</i> <i>boolean</i> <i>dateTime</i> <i>duration</i> <p><i>NamedFeature</i> – This GeneralID represents a NamedFeature as defined in Section 3.5.2.2, Specifying NamedFeatures with GeneralID.%%%</p>
<i>IDUsage</i>	NMOKEN	<p>Usage of the GeneralID. There are no predefined values in XJDF. Values below with "AdsML" prefix are defined in [AdsML].</p> <p>Values include:</p> <p><i>DeviceExternalID</i> – An ID of the resource as defined in the Device namespace. For instance Media catalogs of a press may provide Media identifiers that are different from those defined by the MIS (which are identified with <i>@ExternalID</i> values).</p> <p><i>AdsML:AdBuyer_BookingTransactionID</i> – an ID for the booking transaction that was assigned by a party acting on behalf of the advertiser</p> <p><i>AdsML:AdSeller_BookingTransactionID</i> – an ID for the booking transaction that was assigned by the publisher or a party acting on its behalf</p> <p><i>AdsML:AdBuyer_AdMaterialID</i> – an ID for the artwork that was assigned by a party acting on behalf of the advertiser</p> <p><i>AdsML:AdSeller_AdMaterialID</i> – an ID for the artwork that was assigned by the publisher or a party acting on its behalf.</p> <p><i>LineID</i> – an ID for PrintTalk which associates a PrintTalk//Pricing/Price[@<i>LineID</i> = "someValue"] Element with a JDF Element embedded in PrintTalk, such as PrintTalk//jdf:DeliveryParams/DropItem[GeneralID/@<i>IDUsage</i> = "LineID" and GeneralID/@<i>IDValue</i> = "someValue"].</p>
<i>IDValue</i>	string	Value of the GeneralID. The data type of the value SHALL correspond to GeneralID/@ <i>DataType</i> .

Product Intent Nodes

XJDF Nodes that are created by end customers typically describe only the desired Product rather than manufacturing process details.

These Nodes SHALL have a **@Types** Attribute Value of "*Product*" to indicate that they do not specify any processing

Product elements SHOULD include Product Intent Elements that describe the end results the customer is requesting. If additional process information that is not defined in Product Intent Elements is required, this information SHOULD be provided in **Resource** Elements. The value of **@Types** SHALL remain "*Product*"



3.3.3 Process Nodes

Process nodes contain processing instructions in ResourceSet Elements that are targeted to a specific Device in addition to an optional Product definition in a ProductList and referenced Product Intent elements. The **@Types** Attribute of Process Nodes SHALL NOT contain the token "*Product*" if any additional process type tokens are present.

3.3.3.1 Specifying NamedFeatures with GeneralID

XJDF Nodes MAY contain zero or more **GeneralID**[@Datatype="NamedFeature"] elements to specify global setup definitions. These **GeneralID** elements that are referred to as "NamedFeatures" in this paragraph allow a Controller to define a named set of parameters for Processes that SHALL be executed without defining the details or even the Resources.

Explicitly specified traits SHALL override any implied traits defined by **GeneralID**[@Datatype="NamedFeature"].

XJDF/@Types abstractly specifies the set of Processes to execute, whereas **NamedFeatures** abstractly specifies the set of Resources for the Processes specified in **@Types**.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="J1" Status="Waiting"
      Type="ProcessGroup" JobPartID="ID300" Version="1.4">
    <!--the ResourceLink Elements in the ProcessGroup define the Input
        Resources that are to be available for the ProcessGroup to be
        submitted and the Output Resources that are produced by the ProcessGroup
    -->
    <ResourcePool>
        <DigitalPrintingParams ID="L1" Class="Parameter" Status="Available"/>
        <Media ID="L2" Class="Consumable" Status="Available"/>
        <RunList ID="L8" Class="Parameter" Status="Available"/>
        <Component ID="L3" Class="Quantity" Status="Unavailable"
                  ComponentType="Sheet"/>
        <GatheringParams ID="L4" Class="Parameter" Status="Available"/>
        <Component ID="L5" Class="Quantity" Status="Unavailable"
                  ComponentType="Sheet"/>
        <StitchingParams ID="L6" Class="Parameter" Status="Available"/>
        <Component ID="L7" Class="Quantity" Status="Unavailable"
                  ComponentType="Sheet"/>
    </ResourcePool>
    <ResourceLinkPool>
        <!-- print input media -->
        <MediaLink Usage="Input" rRef="L2"/>
```

```

<!-- gathered output components -->
<ComponentLink Usage="Output" rRef="L7"/>
</ResourceLinkPool>
<JDF ID="J2" Status="Waiting" JobPartID="ID301" Type="DigitalPrinting">
    <ResourceLinkPool>
        <!-- digital printing parameters -->
        <DigitalPrintingParamsLink Usage="Input" rRef="L1"/>
        <!-- input sheets -->
        <MediaLink Usage="Input" rRef="L2"/>
        <RunListLink Usage="Input" rRef="L8"/>
        <!-- printed output components -->
        <ComponentLink Usage="Output" rRef="L3"/>
    </ResourceLinkPool>
</JDF>
<JDF ID="J3" Status="Waiting" JobPartID="ID302" Type="Gathering">
    <ResourceLinkPool>
        <!-- gathering parameters -->
        <GatheringParamsLink Usage="Input" rRef="L4"/>
        <!-- printed output components -->
        <ComponentLink Usage="Input" rRef="L3"/>
        <!-- gathered output components -->
        <ComponentLink Usage="Output" rRef="L5"/>
    </ResourceLinkPool>
</JDF>
<JDF ID="J4" Status="Waiting" JobPartID="ID303" Type="Stitching">
    <ResourceLinkPool>
        <!-- Stitching parameters -->
        <StitchingParamsLink Usage="Input" rRef="L6"/>
        <!-- gathered output components -->
        <ComponentLink Usage="Input" rRef="L5"/>
        <!-- stitched output components -->
        <ComponentLink Usage="Output" rRef="L7"/>
    </ResourceLinkPool>
</JDF>
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="J1" Status="Waiting"
      Type="Combined" Types="DigitalPrinting Gathering Stitching" JobPartID="ID200"
      Version="1.4">
    <ResourceLinkPool>
        <!-- digital printing input RunList -->
        <RunListLink CombinedProcessIndex="0" Usage="Input" rRef="L1"/>
        <!-- digital printing parameters -->
        <DigitalPrintingParamsLink CombinedProcessIndex="0" Usage="Input" rRef="L2"/>
        <!-- gathering parameters -->
        <GatheringParamsLink CombinedProcessIndex="1" Usage="Input" rRef="L3"/>
        <!-- Stitching parameters -->
        <StitchingParamsLink CombinedProcessIndex="2" Usage="Input" rRef="L4"/>
        <!-- input sheets -->
        <MediaLink CombinedProcessIndex="0" Usage="Input" rRef="L5"/>
        <!-- stitched output components -->
        <ComponentLink CombinedProcessIndex="2" Usage="Output" rRef="L6"/>
    </ResourceLinkPool>
    <ResourcePool>
        <RunList ID="L1" Class="Parameter" Status="Available"/>
        <DigitalPrintingParams ID="L2" Class="Parameter" Status="Available"/>
        <GatheringParams ID="L3" Class="Parameter" Status="Available"/>
        <StitchingParams ID="L4" Class="Parameter" Status="Available"/>
        <Media ID="L5" Class="Consumable" Status="Available"/>
    </ResourcePool>

```

```

<Component ID="L6" Class="Quantity" Status="Unavailable"
    ComponentType="Sheet"/>
</ResourcePool>
</JDF>
<JDF ID="ID" xmlns="http://www.CIP4.org/JDFSchema_1_1" Status="Waiting"
    Type="Combined" Types="Collecting Gathering Folding" JobPartID="ID345"
    Version="1.4">
<ResourcePool>
    <GatheringParams ID="gp1" Class="Parameter" Status="Available"/>
    <FoldingParams ID="fp1" Class="Parameter" Status="Available"/>
    <Component ID="in1" Class="Quantity" Status="Available" ComponentType="Sheet"/>
    <Component ID="in2" Class="Quantity" Status="Available" ComponentType="Sheet"/>
    <Component ID="in3" Class="Quantity" Status="Available" ComponentType="Sheet"/>
    <Component ID="ex1" Class="Quantity" Status="Unavailable" ComponentType="Sheet"
        PipeProtocol="Internal" PipeID="ex1"/>
    <Component ID="ex2" Class="Quantity" Status="Unavailable" ComponentType="Sheet"
        PipeProtocol="Internal" PipeID="ex2"/>
    <Component ID="Out" Class="Quantity" Status="Unavailable"
        ComponentType="Sheet"/>
</ResourcePool>
<ResourceLinkPool>
    <GatheringParamsLink Usage="Input" rRef="gp1"/>
    <FoldingParamsLink Usage="Input" rRef="fp1"/>
    <ComponentLink CombinedProcessIndex="0" Usage="Input" rRef="in1"/>
    <ComponentLink CombinedProcessIndex="0" Usage="Input" rRef="in2"/>
    <ComponentLink CombinedProcessIndex="2" Usage="Input" rRef="in3"/>
    <ComponentLink CombinedProcessIndex="0" Usage="Output" rRef="ex1"/>
    <ComponentLink CombinedProcessIndex="2" Usage="Output" rRef="ex2"/>
    <ComponentLink CombinedProcessIndex="1" Usage="Input" rRef="ex1"/>
    <ComponentLink CombinedProcessIndex="1" Usage="Input" rRef="ex2"/>
    <ComponentLink CombinedProcessIndex="1" Usage="Output" rRef="Out"/>
</ResourceLinkPool>
</JDF>

```

3.4 ProductList

The **ProductList** specifies a list of products and sub products from the print buyers point of view. For more details on product intent, see Chapter 7, “Product Intents”.

Table 3-4: ProductList Element

Name	Data Type	Description
Product	Element	Each Product element in this list represents a Product or part of a product with unique properties such as substrate, colors or size.

3.5 ResourceSet

One **ResourceSet** describes a set of one or more **Resource** elements that are logically grouped together. Thus two **ResourceSet** elements with the same **@Name** and **@Usage** MAY coexist in an XJDF if they define logically separated entities. In this case **ResourceSet/@ProcessUsage** SHALL be specified and different. For Instance **NodeInfo** MAY be defined based on End Customer scheduling requirements and partitioned by **@ProductPart** and the production scheduling SHOULD be defined in a separate **ResourceSet**.

The **Resource** Element is described in the Section 8.1, “Resource”.

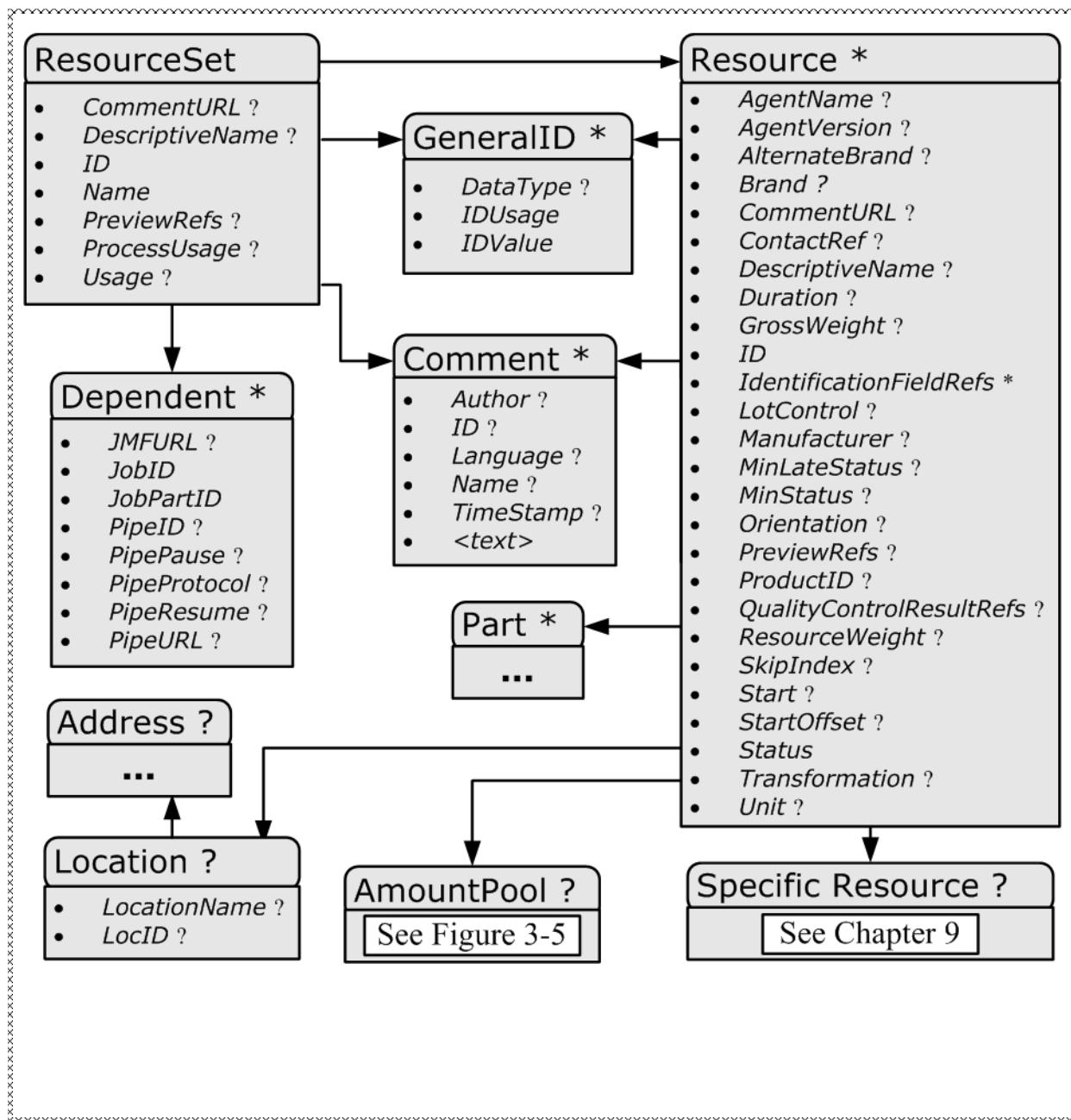
Table 3-5: ResourceSet Element

Name	Data Type	Description
<i>CommentURL?</i>	URL	URL to an external, human-readable description of the ResourceSet .
<i>DescriptiveName?</i>	string	Human-readable descriptive name of the ResourceSet .
<i>ID?</i>	ID	Identifier of the ResourceSet .
<i>Name</i>	string	Type of the ResourceSet . A list of valid types is specified in Chapter 8, ‘Resources’
<i>ProcessUsage?</i>	NMTOKEN	<p><i>ProcessUsage</i> identifies the context of a Resource if multiple Resource elements of the same type MAY be supplied. For example, <i>@ProcessUsage</i> SHALL be provided when two RunList Resources - Marks and Document are used in Impositioning . <i>@ProcessUsage</i> MAY also be used to define the Individual Process Step from XJDF/ <i>@Types</i> that a given ResourceSet applies to.</p> <p>Values include those specified in the appropriate Process descriptions in Chapter 6, “Processes”.</p> <p>Note: ICS documents MAY define additional values for <i>@ProcessUsage</i>.</p>
<i>Unit?</i>	NMTOKEN	<p>Unit of measurement for the values of <i>@Amount</i>, <i>@AmountProduced</i> and <i>@AmountRequired</i>.</p> <p>Values include those from: Table 1-8, “Units Used in XJDF”.</p> <p>Note: that it is strongly discouraged to specify units other than those that are defined in Table 1-8, “Units Used in XJDF”.</p>
<i>Usage?</i>	enumeration	<p>Resource usage within this XJDF Document.</p> <p>If no <i>@Usage</i> is specified, the ResourceSet or its child Resource elements SHALL be referenced from another Resource or sub-element thereof.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Input</i> – The ResourceSet is an input that is consumed by the XJDF. <i>Output</i> – The ResourceSet is an output that is produced by the XJDF
<i>Comment*</i>	element	Any human-readable text that relates to the ResourceSet .
<i>Dependent*</i>	element	Reference to an XJDF that produces this input ResourceSet or consumes this output ResourceSet .
<i>GeneralID*</i>	element	Additional identifiers related to the ResourceSet .
<i>Resource*</i>	element	List of Resource Elements.

3.5.1 Structure Diagram of ResourceSet

Figure 3-2 shows the structure of the **ResourceSet**. Arrows point to child elements.

Figure 3-2: ResourceSet – a Diagram of its Structure



3.5.2 Element: Dependent

A Dependent Element SHALL reference an XJDF that produces an input ResourceSet or consumes an output ResourceSet of the Dependent element's parent ResourceSet. Dependent elements also provide all pipe control information. See Section , “Overlapping Processing Using Pipes” on page 184 and Section 5.7.1, PipeControl.

Table 3-6: Dependent Element

Name	Data Type	Description
JMFURL?	URL	URL of a processor that has knowledge of the referenced process. This MAY be either a Device, Controller or MIS. The URL may provide additional information through XJMF.
JobID	NMTOKEN	@JobID of the referenced process.
JobPartID	NMTOKEN	@JobPartID of the referenced process.
PipeID?	NMTOKEN	If this Attribute exists, the Resource is a pipe. @PipeID is used by XJMF pipe-control Messages to identify the pipe. For more information, see Section , “Overlapping Processing Using Pipes”.
PipePartIDKeys?	enumerations	Defines the list of valid partition keys of a dynamic pipe for a resource such as RunList with @Automation = "Dynamic". Resource/Part of any PipeControl message SHALL NOT contain partition keys other than those specified in either @PipePartIDKeys or the respective ../ Resource/Part . The contents of @PipePartIDKeys SHALL NOT contain any enumeration values that are specified in any ../ Resource/Part
PipePause?	double	Attribute for controlling the pausing of a Process if the Resource amount in the pipe buffer passes the specified value. For details on using @PipePause, see Section , “Overlapping Processing Using Pipes”.
PipeProtocol?	NMTOKEN	Defines the protocol use for pipe handling. "JMF" and "Internal" are the only non-proprietary piping protocols that are supported. Proprietary pipe protocols MAY be specified in addition to those defined below but will not necessarily be interoperable. Values include: IdentificationField – The pipe data is provided by barcodes that are defined in IdentificationField elements. JMF – XJMF-based PipeControl Messages. The sequence of pipe initialization is undefined. See next two values: "JMFPush" and "JMFPull". JMFPush – XJMF based PipeControl protocol. The producing Device initiates the protocol. JMFPull – XJMF based PipeControl protocol. The consuming Device initiates the protocol. None – No pipe support.
PipeResume?	double	Attribute for controlling the resumption of a Process if the Resource amount in the pipe buffer passes the specified value. For details on using @PipeResume, see Section , “Overlapping Processing Using Pipes”.

3.6 Subelements of Resource

This section defines an Element for each type of **Resource**.

3.6.1 Location

Resource elements can be stored at multiple, distributed locations. This is specified by including a **Location** Element in the **Resource** Element. See Section 8.1.1, “Location”. A @Location Partition Key is provided to define multiple locations of one **Resource**. The Partition Key carries no semantic meaning and does not by itself define the name of a location.

Example 3-1: ExposedMedia with Location Elements

The following example describes a set of plates that are distributed over two locations. (Note: See Section C.4, “Input Tray and Output Bin Names” on page 1175 for additional detail on locating **Resource**.

TBD 2.x Example.

Example 3-2: Media with Location Elements

The following example describes two different Media in the top and bottom tray of an **Interpreting** Process. The Media is selected for the cover and inside pages respectively.

TBD 2.x Example.

3.6.2 Part

The optional **Part** elements define the context in which the individual **Resource** is used. In our printing plate example, this would typically include the **@SheetName**, **@Side** and **@Separation** that the printing plate is for.

Resource partitions are uniquely identified by the **Resource/Part** elements. If multiple **Part** Elements are specified within one **Resource**, the **Resource** specifies one entity that applies to both parts.

The following table lists the content of a **Part** Element, which contains a set of Attributes that have a well described meaning.

Table 3-7: Part Element (Sheet 1 of 6)

Name	Data Type	Description
<i>BinderySignatureID?</i>	NMTOKEN	Master Identifier of a BinderySignature .
<i>BlockName?</i>	NMTOKEN	Identifies a CutBlock from a Cutting Process. The value of this Attribute SHALL match the value of the @BlockName Attribute of a CutBlock .
<i>BundleItemIndex?</i>	IntegerRange	The @BundleItemIndex Attribute selects a set of BundleItem Elements from a Component Resource.
<i>DropID?</i>	NMTOKEN	Identifier of an individual drop within a Delivery . A drop represents multiple items being delivered to one address at one point in time.
<i>DeliveryUnit0?</i>	NMTOKEN	Specifies a hierarchical manifest of delivery packages where @DeliveryUnit0 specifies the most granular bundle. @DeliveryUnit<N+1> specifies the next most granular bundle in packing after @DeliveryUnit<N> . Bundles can be packaged with varying numbers of products. @DeliveryUnit<N+1> SHALL occur before @DeliveryUnit<N> in @PartIDKeys . Note that N is a placeholder for the values 0 through 9.
<i>DeliveryUnit1?</i>	NMTOKEN	See @DeliveryUnit0 .
<i>DeliveryUnit2?</i>	NMTOKEN	See @DeliveryUnit0 .
<i>DeliveryUnit3?</i>	NMTOKEN	See @DeliveryUnit0 .
<i>DeliveryUnit4?</i>	NMTOKEN	See @DeliveryUnit0 .
<i>DeliveryUnit5?</i>	NMTOKEN	See @DeliveryUnit0 .
<i>DeliveryUnit6?</i>	NMTOKEN	See @DeliveryUnit0 .
<i>DeliveryUnit7?</i>	NMTOKEN	See @DeliveryUnit0 .
<i>DeliveryUnit8?</i>	NMTOKEN	See @DeliveryUnit0 .

Table 3-7: Part Element (Sheet 2 of 6)

Name	Data Type	Description
<i>DeliveryUnit9?</i>	NMTOKEN	See @ <i>DeliveryUnit0</i> .
<i>DocIndex?</i>	IntegerRange	The @ <i>DocIndex</i> Attribute selects a set of logical Instance Documents from a RunList Resource.
<i>DocRunIndex?</i>	IntegerRange	The @ <i>DocRunIndex</i> Attribute selects a set of logical pages from Instance Documents of a RunList Resource. For example, @ <i>DocRunIndex</i> = "0 -1" specifies the first and last page of every copy of every selected Instance Document (assuming that additional Partitioning using <i>DocIndex</i> is not also specified). Specifying @ <i>DocRunIndex</i> does not modify the index of a RunList entry and therefore does not reposition pages on a Layout .
<i>DocSheetIndex?</i>	IntegerRange	The @ <i>DocSheetIndex</i> Attribute selects a set of logical Sheets from individual Instance Documents. For example @ <i>DocSheetIndex</i> = "0 -1" specifies the first and last Sheet of every selected copy of every Instance Document (assuming that additional Partitioning using <i>DocIndex</i> is not also specified). Specifying @ <i>DocSheetIndex</i> does not modify the index of a RunList entry and therefore does not reposition pages on a Layout .
<i>DocTags?</i>	RegExp	List of tags of documents in a multi-document RunList . Used to Partition Resources that are linked from Processes that also have a RunList as input. The Partition is selected if the implied value (i.e., from the PDL) of the document in the RunList matches any of the entries in @ <i>DocTags</i> Note that being a multi-set RunList implies being a multi-document RunList as well.
<i>Location?</i>	NMTOKEN	Name of the location (e.g., in MIS). This part key allows to describe distributed Resources. Note that this name does not define the location by itself. See Section 3.6.1, “Location” for details on specifying locations. Values include those from: Table C-11, “Input Tray and Output Bin Names” on page 1175. Note: the specified values are for printer locations.
<i>LotID?</i>	NMTOKEN	Identifier of the Lot of a resource in a lot controlled environment.
<i>Metadata0?</i>	RegExp	Metadata extracted from a PDL using RunList/MetadataMap Elements. See Section 10.21, “MetadataMap” on page 1090.
<i>Metadata1?</i>	RegExp	See @ <i>Metadata0</i> .
<i>Metadata2?</i>	RegExp	See @ <i>Metadata0</i> .
<i>Metadata3?</i>	RegExp	See @ <i>Metadata0</i> .
<i>Metadata4?</i>	RegExp	See @ <i>Metadata0</i> .
<i>Metadata5?</i>	RegExp	See @ <i>Metadata0</i> .
<i>Metadata6?</i>	RegExp	See @ <i>Metadata0</i> .
<i>Metadata7?</i>	RegExp	See @ <i>Metadata0</i> .
<i>Metadata8?</i>	RegExp	See @ <i>Metadata0</i> .

Table 3-7: Part Element (Sheet 3 of 6)

Name	Data Type	Description
<i>Metadata9?</i>	RegExp	See @ <i>Metadata0</i> .
<i>Option?</i>	NMTOKEN	Generic option that MAY be semantic free.
<i>PageNumber?</i>	IntegerRange	Page number in a Component or document (e.g., FileSpec) that is not described as a RunList). References an index in a Content where Content/@ContentType = "Page" .
<i>PageTags?</i>	RegExp	List of tags of pages in a multi-page RunList . Used to Partition Resources that are linked from Processes that also have a RunList as input. The Partition is selected if the implied value (i.e., from the PDL) of the page in the RunList matches any of the entries in @ <i>PageTags</i> .
<i>PartVersion?</i>	NMTOKEN	Version identifier (e.g., the language version of a catalog).
<i>PreviewType?</i>	enumeration	<p>Preview specifies the type and usage of a Preview. @PreviewType SHALL NOT be specified for Resources other than Preview or PreviewGenerationParams.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Animation</i> – animated previews for 3D display. <i>Identification</i> – Preview is used as a visual help to identify one or more products, e.g. on a gang form <i>SeparatedThumbnail</i> – Very low resolution separated preview. <i>Separation</i> – Separated preview in medium resolution. <i>SeparationRaw</i> – Separated preview in medium resolution with no compensation. <i>Static3D</i> – static 3D model. <i>Thumbnail</i> – Very low resolution RGB preview. <i>Viewable</i> – RGB preview in medium resolution.
<i>PrintCondition?</i>	NMTOKEN	Name of the printing condition. Used to differentiate specific color properties of a colorant; i.e., how is the printed color result specific to media, screening, etc.
<i>ProductPart?</i>	NMTOKEN	References the Product/@ID that this Part applies to.
<i>QualityMeasurement?</i>	NMTOKEN	Identifier of an individual quality measurement in a QualityControl process.
<i>Run?</i>	NMTOKEN	The @ <i>Run</i> Attribute selects an individual RunList Partition from a RunList Resource.

Table 3-7: Part Element (Sheet 4 of 6)

Name	Data Type	Description
<i>RunIndex</i> ?	IntegerRange	The <i>@RunIndex</i> Attribute selects a set of logical pages from a RunList Resource in a manner that is independent from the internal structure of the RunList . It contains one or more pairs of integers. Each pair of integers is a range. If the two integers in a pair are the same, the pair represents a single integer value. Each integer is separated by space character. For example, <i>@RunIndex</i> = "2 5 8 8 10 10 22 ~ -1". Negative numbers reference pages from the back of a file in base-1 counting. In other words, -1 is the last page, -2 the second to last, etc. Thus <i>@RunIndex</i> = "0 -1" refers to a complete range of pages, from first to last. The index always refers to entries of the entire RunList and SHALL NOT be modified if only a part of the RunList is spawned. Specifying <i>@RunIndex</i> does not modify the index of a RunList entry and therefore does not reposition pages on a Layout .
<i>RunPageRange</i> ?	IntegerRange	Used when splitting RunList Resources into larger chunks that are not yet based on indices of Content where Content / <i>@ContentType</i> = "Page".
<i>RunSet</i> ?	NMOKEN	Generic group of Elements in a RunList . If Partitioning a RunList by <i>@RunSet</i> and <i>@Run</i> , then <i>@RunSet</i> SHOULD be specified closer to the root.
<i>RunTags</i> ?	RegExp	List of names in a named RunList . Used to Partition Resources that are linked from Processes that also have a RunList as input when the sequence of the RunList is undefined. The Partition is selected if the explicit or implied (e.g., from the PDL) value of <i>@RunTag</i> of the RunList matches any of the entries in <i>@RunTags</i> . Note: the difference between <i>@RunTags</i> and <i>@PageTags</i> , <i>@DocTags</i> or <i>@SetTags</i> . <i>@PageTags</i> is used to identify classes of individual pages having differing XJDF parameterization. Similarly, <i>@DocTags</i> is used to identify classes of individual documents and <i>@SetTags</i> is used to identify classes of individual sets each having differing XJDF parameterization. <i>@RunTags</i> is used to identify collections of pages, often thought of as a document or a piece of a document, but not limited to that. Also, <i>@RunTags</i> MAY be explicitly set for an entire RunList by use of the <i>@RunTag</i> Attribute. The <i>@SetTags</i> , <i>@DocTags</i> and <i>@PageTags</i> Partition Keys are always set implicitly and always refer to the granularity within a RunList implied by their names.

Table 3-7: Part Element (Sheet 5 of 6)

Name	Data Type	Description
<i>Separation</i> ?	NMTOKEN	<p>Identifies the separation name.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Composite</i> – Non-separated Resource. <i>Separated</i> – The Resource is separated, but the separation definition is handled internally by the Resource, such as a PDF file that contains SeparationInfo dictionaries. <i>Cyan</i> – Process color. <i>Magenta</i> – Process color. <i>Yellow</i> – Process color. <i>Black</i> – Process color. <i>Red</i> – Additional process color. <i>Green</i> – Additional process color. <i>Blue</i> – Additional process color. <i>Orange</i> – Additional process color. <i>Spot</i> – Generic spot color. Used when the exact nature of the spot color is unknown. <i>Varnish</i> – Varnish. <i>none</i> – explicit reference to a skipped module (i.e., no separation). Note that "<i>none</i>" is spelled in lower case so that its value is identical to the pre-defined separation "<i>none</i>" in [PDF1.6]. <p>Note: Other values specify spot colors.</p>
<i>SetDocIndex</i> ?	IntegerRange	The @SetDocIndex Attribute selects a set of logical Instance Documents from Instance Document Sets of a RunList Resource. For example, @SetDocIndex = "0 -1" specifies the first and last page of every copy of every selected Instance Document Set. The index always refers to entries of the entire RunList and SHALL NOT be modified if only a part of the RunList is spawned. Specifying @SetDocIndex does not modify the index of a RunList entry and therefore does not reposition pages on a Layout .
<i>SetIndex</i> ?	IntegerRange	The @SetIndex Attribute selects a set of logical Instance Document Sets from a RunList Resource. The index always refers to entries of the entire RunList and SHALL NOT be modified if only a part of the RunList is spawned. Specifying @SetIndex does not modify the index of a RunList entry and therefore does not reposition pages on a Layout .
<i>SetRunIndex</i> ?	IntegerRange	The @SetRunIndex Attribute selects a set of logical pages from in-stance Document Sets of a RunList Resource. For example, @SetRunIndex = "0 -1" specifies the first and last page of every copy of every selected Instance Document Set. The index always refers to entries of the entire RunList and SHALL NOT be modified if only a part of the RunList is spawned. Specifying @SetRunIndex does not modify the index of a RunList entry and therefore does not reposition pages on a Layout .

Table 3-7: Part Element (Sheet 6 of 6)

Name	Data Type	Description
<i>SetSheetIndex?</i>	IntegerRange	The <i>@SetSheetIndex</i> Attribute selects a set of logical Sheets from individual sets of Instance Documents. For example <i>@SetSheetIndex = "0 -1"</i> specifies the first and last Sheet of every selected copy of every Set. The index always refers to entries of the entire RunList and SHALL NOT be modified if only a part of the RunList is spawned. Specifying <i>@SetSheetIndex</i> does not modify the index of a RunList entry and therefore does not reposition pages on a Layout .
<i>SetTags?</i>	RegExp	List of tags of pages in a multi-set RunList . Used to Partition Resources that are linked from Processes that also have a RunList as input. The Partition is selected if the implied value (i.e., from the PDL) of the set in the RunList matches any of the entries in <i>@SetTags</i> .
<i>SheetIndex?</i>	IntegerRange	The <i>@SheetIndex</i> Attribute selects a set of logical Sheets from a RunList Resource either implicitly or explicitly Partitioned by <i>@SheetIndex</i> . <i>@SheetIndex</i> is only valid when a RunList is describing sheet/surfaces.
<i>SheetName?</i>	NMTOKEN	A string that uniquely identifies each Sheet.
<i>Side?</i>	enumeration	Denotes the side of the Sheet. If <i>@Side</i> is specified, the Part Element refers to one or both surfaces of the Sheet. In case of Web Printing, " <i>Front</i> " is a synonym for the upper side and " <i>Back</i> " for the down side of the Web. Allowed values are: <i>Front</i> <i>Back</i>
<i>StationName?</i>	NMTOKEN	The name of the 1-up design in a DieLayout .
<i>TileID?</i>	XYPair	XYPair of integer values that identifies the tile. Tiles are identified by their X and Y indexes. Values are zero-based and expressed in the PS coordinate system. So " <i>0 0</i> " is the lower left tile and " <i>1 0</i> " is the tile next to it on the right. Tile Resources are described in detail in the Section 8.102, "Tile".
<i>TransferCurveName?</i>	NMTOKEN	<i>@TransferCurveName</i> SHALL specify the destination system that the TransferCurve SHALL apply to. Allowed values are: <i>Film</i> – The transformation from the Layout system to the " <i>Film</i> ". <i>Plate</i> – The transformation from the Layout system to the " <i>Plate</i> ". <i>Press</i> – The transformation from the Layout system to the " <i>Press</i> ". <i>Paper</i> – The transformation from the Layout system to the " <i>Paper</i> ". <i>Proof</i> – The transformation from the Layout system to the " <i>Proof</i> ".
<i>WebName?</i>	NMTOKEN	A string that uniquely identifies each Web.

3.6.2.1 Partitioned Subsets of ResourceSets

In many cases, a set of similar **Resource** elements — such as separation films, plates or **RunList** elements — needs to be specified. When this occurs, it is convenient to define one **Resource** element that describes the complete set and allows individual subsets to be referenced.

In other cases, there can be a need to change some Attribute of a **Resource** for some subset of the processing to be done by a Device. For instance, when printing a document using **DigitalPrinting**, it would be a common application to change the dimensions of the media to be selected based on the actual media box changes in a PDF file.

Printing workflows contain a number of Processes that are repeated over a potentially large number of individual files, Sheets, surfaces or separations. Sets of resources are described by a **ResourceSet** with multiple **Resource** elements. An individual **Resource** SHALL be referenced by referencing **Resource/@ID**. The **ResourceSet** SHALL be referenced by referencing **ResourceSet/@ID**.

In some cases the partitioning structure of partitioned **ResourceSets** is not explicitly required because the **Resource** elements are individually referenced from other elements. In this case the **Part** Element NEED NOT be specified, even if there are multiple **Resource** elements in one **ResourceSet**.

Unless otherwise specified, an **@IDREF** or **@IDREFS** will refer to an individual **Resource** rather than an entire **ResourceSet**.

3.6.2.2 Selecting a Partition

A matching partition for a given set of partition keys is selected by iterating the **Resource** elements of the respective **ResourceSet** from top to bottom. If any of the **Resource/Part** elements has no mismatching attributes, that **Resource** is selected and the iteration is completed. Note that there NEED NOT be a match for any given Partition. Note: A **Resource/Part** with no attributes always matches.

3.6.2.3 Referencing Multiple Resources of the Same Type

Some Processes (e.g., **Collecting**, **Gathering**) allow multiple Input Resources of the same type. These multiple Input Resources SHALL be represented by multiple individual elements. If ordering is not explicitly defined, e.g. by specification of an Assembly, then the order of the **ResourceSet** elements in the XJDF Node defines said ordering.

3.6.2.4 Multiple Part Elements in One Resource

A **ResourceSet** MAY contain one or more **Resource** elements which MAY respectively contain 0 or more **Part** elements. Each **Resource** represents one entity, regardless of the number of **Part** elements. If a **Resource** contains more than one **Part** element, this **Resource** is applicable to any of the contained **Part** elements. For instance a set of plates for a versioned CMYK sheet with black change for English and French versions could have 5 plates. The C, M and Y would each contain 2 **Part** elements with both English and French whereas the two individual **Resource** elements for the K Plates would contain the respective **@PartVersions**.

Example 3-3: Versioned Set Of Plates with Multiple Part Elements

```
<ResourceSet Name="ExposedMedia">
    <!-- 3 Common Plates for English and French -->
    <Resource>
        <Part Separation="Cyan" PartVersion="English"/>
        <Part Separation="Cyan" PartVersion="French"/>
        <ExposedMedia .../>
    </Resource>
    <Resource>
        <Part Separation="Magenta" PartVersion="English"/>
        <Part Separation="Magenta" PartVersion="French"/>
        <ExposedMedia .../>
    </Resource>
    <Resource>
        <Part Separation="Yellow" PartVersion="English"/>
        <Part Separation="Yellow" PartVersion="French"/>
    </Resource>
```

```

<ExposedMedia .../>
</Resource>
<!-- Specific Black Plate for English --&gt;
&lt;Resource&gt;
  &lt;Part Separation="Black" PartVersion="English"/&gt;
  &lt;ExposedMedia .../&gt;
&lt;/Resource&gt;
<!-- Specific Black Plate for French --&gt;
&lt;Resource&gt;
  &lt;Part Separation="Black" PartVersion="French"/&gt;
  &lt;ExposedMedia .../&gt;
&lt;/Resource&gt;
&lt;/ResourceSet&gt;
</pre>

```

3.6.3 AmountPool

Whereas **Resource/Part** identifies the context of **Resource** that the Process is consuming or producing, **AmountPool** is a container for the amount-related metadata of the **Resource**. **AmountPool/PartAmount/Part** SHALL refer to existing Partitions or non-existing sub-partitions of existing partitions that are implicitly or explicitly referred to by **Resource/Part**. For instance, if a **Resource** refers to a partition with `@SheetName="Sheet1"`, **AmountPool/PartAmount** MAY refer to Sheet1 or any existing or non-existing child of Sheet1, but NOT to Sheet2 or any existing or non-existing child of Sheet2.

The following table lists the generic contents of an **AmountPool** Element. Further parameters of the **AmountPool** are described in the sections below.

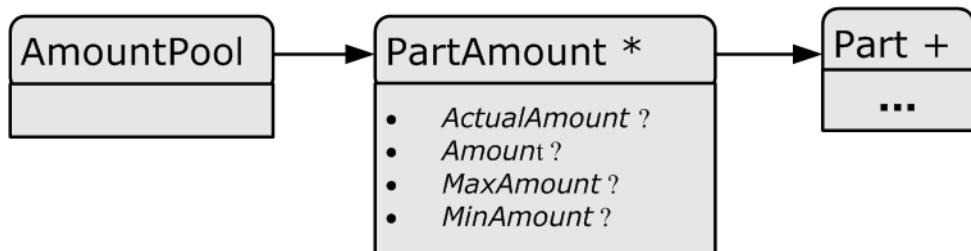
Table 3-8: AmountPool Element

Name	Data Type	Description
PartAmount *	element	Element that defines the amounts and pipe parameters for a Partitioned Resource.

3.6.3.1 Structure Diagram of AmountPool

Figure 3-3 shows the structure of the **AmountPool**. Arrows point to child elements.

Figure 3-3: AmountPool – a Diagram of its Structure



3.6.3.2 Element: PartAmount

The following table lists the contents of a **PartAmount** Element.

Table 3-9: PartAmount Element

Name	Data Type	Description
ActualAmount?	double	Total amount of the Resource that has been produced (in a Resource with <i>@Usage</i> = "Output") or consumed (in a Resource with <i>@Usage</i> = "Input") by this Node in every execution. For details see Section 3.6.4, "Resource Amount".
Amount?	double	Good Amount in units defined in ResourceSet / <i>@Unit</i> or implied by Table 1-8, "Units Used in XJDF". For an ResourceSet with a <i>@Usage</i> of "Input", <i>@Amount</i> specifies the amount of the ResourceSet that is needed by the Process. For an ResourceSet with a <i>@Usage</i> of "Output", <i>@Amount</i> specifies the amount of the ResourceSet that SHALL be produced by the Process.
MaxAmount?	double	Defines the planned <i>@Amount</i> including the maximum overage.
MinAmount?	double	Defines the planned <i>@Amount</i> including the maximum underage that the Customer is willing to accept.
ModuleIDs?	NMTOKENS	Specifies the module or modules where the waste was produced.
Waste?	double	Waste amount in units defined in ResourceSet / <i>@Unit</i> or implied by Table 1-8, "Units Used in XJDF". For a ResourceSet with a <i>@Usage</i> of "Input", <i>@Waste</i> specifies the amount of the ResourceSet that is needed by the Process. For an ResourceSet with a <i>@Usage</i> of "Output", <i>@Waste</i> specifies the amount of the ResourceSet that SHALL be produced by the Process.
WasteDetails?	NMTOKEN	Specifies additional details about how the waste was produced. See Table 3-10, "WasteDetails Attribute Values" for suggested values.
Part +	element	Part specifies the selected parts that the PartAmount is valid for. If the parent AmountPool is specified in a Resource element that also contains Part elements, then these PartAmount/Part elements SHALL NOT include any partition keys that are already specified in the parent Resource /Part.

— Attribute: WasteDetails**Table 3-10: WasteDetails Attribute Values (Sheet 1 of 2)**

Value	Description
Waste	General waste.
Rejected	Rejected in an Approval process
Overrun	Excess Component Resource(s) that were produced by running the Device after the specified amount has been produced.
xxxWaste	Like "Waste" above, but where "xxx" can be the name of any XJDF Process (e.g., "FeedingWaste", "TrimmingWaste", etc.). In the case of a Combined Process or Process Group, the name of the last XJDF Process in the Process chain is used.

Table 3-10: WasteDetails Attribute Values (Sheet 2 of 2)

Value	Description
AuxiliarySheet	This Partition identifies Media that was consumed as specified by InsertSheet/@SheetType = "AccountingSheet", "ErrorSheet", "JobSheet" or "SeparatorSheet".
BindingQualityTestFailed	Failed binding quality test. The Component Resource(s) with this @ConditionWasteDetails belong to the batch of Component Resource(s) that did not pass the test.
BindingQualityTestPassed	Passed binding quality test. The Component Resource(s) with this @WasteDetails belong to the batch of Component Resource(s) that passed the test but were not destroyed in the Process.
BindingQualityTestWaste	Passed binding quality test. The Component Element(s) with this @WasteDetails belong to the batch of Component Element(s) that passed the test but were destroyed in the Process.
CaliperWaste	Waste by caliper on gathering / collecting.
DoubleFeedWaste	Waste by double feeds on feeders.
IncorrectComponentWaste	Waste by the attempted use of an incorrect components (e.g., on a feeder).
BadFeedWaste	Waste caused by a bad feed.
ObliqueSheetWaste	Waste by oblique Sheets on gathering / collecting chains.
PaperJamWaste	Waste by paper or other media jam.
Reusable	Waste to be used for setup in the next process.
WhitePaperWaste	White paper waste.

3.6.3.3

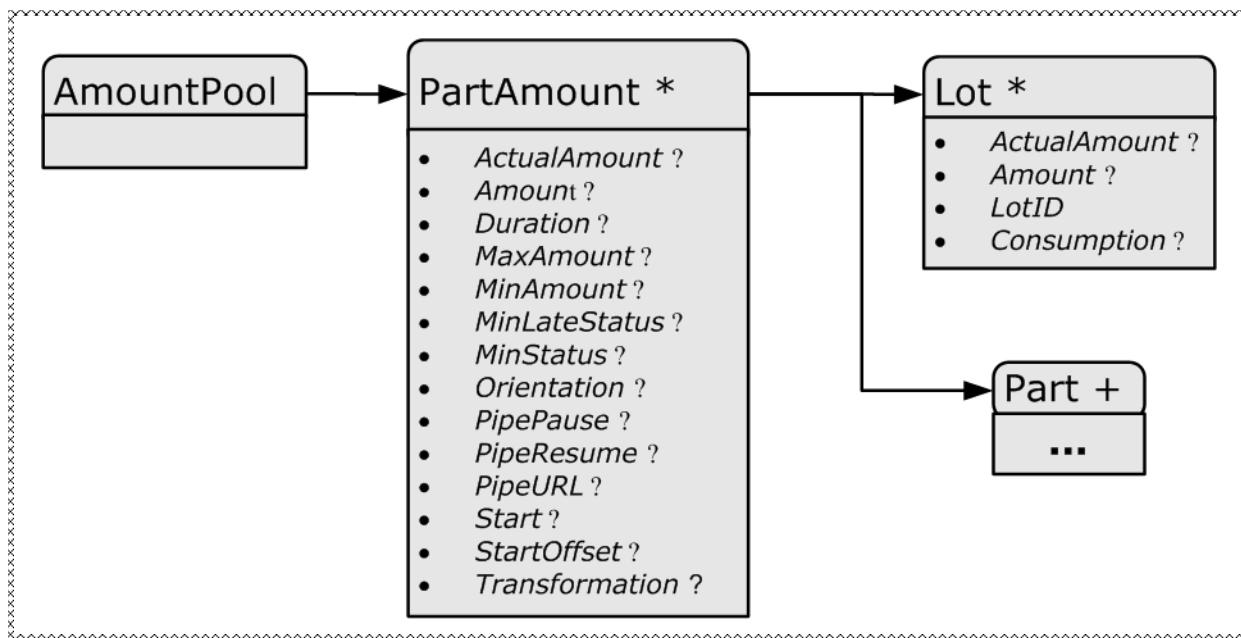
3.6.3.4

```

<ResourcePool>
    <Employee Class="Implementation" ID="L1" Status="Available"
        PersonalID="007">
        <Person FamilyName="Smith" JobTitle="Press Operator"/>
    </Employee>
</ResourcePool>
<ResourceLinkPool>
    <EmployeeLink Usage="Input" rRef="L1"/>
</ResourceLinkPool>

```

Figure 3-4: AmountPool – a Diagram of its Structure



```

<ResourcePool>
  <Ink Brand="NoName" Class="Consumable" ID="Link0015"
      PartIDKeys="Separation" Status="Available">
    <Ink ColorName="Cyan" Separation="Cyan"/>
    <Ink ColorName="Magenta" Separation="Magenta"/>
    <Ink ColorName="Yellow" Separation="Yellow"/>
    <Ink ColorName="Black" Separation="Black"/>
    <Ink ColorName="Heidelberg Spot Blau"
        Separation="Heidelberg Spot Blau"/>
  </Ink>
</ResourcePool>
<ResourceLinkPool>
  <InkLink Usage="Input" rRef="Link0015">
    <AmountPool>
      <PartAmount Amount="1000">
        <Part Separation="Cyan"/>
      </PartAmount>
      <PartAmount Amount="1200">
        <Part Separation="Magenta"/>
      </PartAmount>
      <PartAmount Amount="700">
        <Part Separation="Yellow"/>
      </PartAmount>
      <PartAmount Amount="3000">
        <Part Separation="Black"/>
      </PartAmount>
      <PartAmount Amount="300">
        <Part Separation="Heidelberg Spot Blau"/>
      </PartAmount>
    </AmountPool>
  </InkLink>
</ResourceLinkPool>

```

3.6.4 Resource Amount

The **@Amount** Attribute of an **AmountPool/PartAmount** Element contains the planned amount of a given Resource. The units used in the **@Amount** Attribute of a **Resource** Element is defined by the unit of the **Resource** Element to which the link refers.

Note that for Resources which are the output of Processes, AmountPool/PartAmount/@Amount determines the quantity of the Resource to be produced. For example, in a **DigitalPrinting** Process that included a **RunList** as its input with 16 pages to be printed and a **Component** to its output, @Amount would indicate the number of copies of those 16 pages that the Process would produce.

3.6.4.1 Specifying Amount for a Partially-Completed Process

A Process can be interrupted before the requested amount of output has been produced. When the Job is resent from the Controller to the Device, the Controller SHALL specify only the remaining *@Amount* that the Device SHALL produce in the resent job run.

3.6.5 Identification of Physical Resources

MIS systems frequently include functionality for managing inventory. Many Resources that are consumed by production Processes are things that are tracked for inventory management purposes. This allows estimating the value of the Resources, ensuring that sufficient quantities are on hand, and tracking which specific Resources are used in production of which Jobs. At the most basic level, these Resources MAY be identified in XJDF with `Resource/@ExternalID`.

Some MIS systems track these Resources at lower levels of detail, tracking individual Resource lots. An example of this might include tracking the individual rolls or boxes of paper. While it is theoretically possible to track individual Resource lots using a single identifier, many MIS users choose to track them with more than one identifier. Examples of some of these identifiers include roll numbers, lot numbers, purchase order numbers, receipt dates.

Because the required identifiers may be different from site to site, or even from one type Resource to another, it is not possible to track these Resources with multiple identifiers using **XJDF**. Conveying the identification requirements to Devices would be too complex. Instead, a single identifier is used in **XJDF**. In cases where multiple identifiers are normally used, the MIS SHALL generate a unique identifier for each unique Resource lot. This unique identifier SHALL then be mapped back to the correct unique Resource lot by the MIS.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" Type="ConventionalPrinting"
      Status="Completed" ID="ID100" JobPartID="ID345" Version="1.4">
  <ResourcePool>
    <Media ID="RI007" Class="Consumable" ProductID="3002" Status="Unavailable"
          Brand="Coated Roll Stock" Dimension="2520 8640000"
          MediaType="Paper" Thickness="36"/>
    <ConventionalPrintingParams ID="RI008" Class="Parameter" Status="Available"/>
    <Component ID="RI009" Class="Quantity" Status="Unavailable"
               ComponentType="Sheet"/>
  </ResourcePool>
  <ResourceLinkPool>
    <MediaLink rRef="RI007" Amount="9800" ActualAmount="9703" Usage="Input">
      <Lot ActualAmount="5250" Consumption="Full"
           LotID="LN1040788312RN2005091-04"/>
      <Lot ActualAmount="4453" Consumption="Partial"
           LotID="LN1040788339RN2005091-01"/>
    </MediaLink>
    <ConventionalPrintingParamsLink rRef="RI008" Usage="Input"/>
    <ComponentLink rRef="RI009" Usage="Output"/>
  </ResourceLinkPool>
</JDF>
```

```

    Status="Available">
  <Comment Name="foo">bar</Comment>
  <Media Location="desk"/>
  <Media Location="drawer"/>
</Media>
<Layout Class="Parameter" ID="Sheet" Status="Available">
  <MediaRef rRef="MediaID"/>
</Layout>
<Layout Class="Parameter" ID="Sheet" Status="Available">
  <Media Dimension="72 72">
    <Comment Name="foo">bar</Comment>
  </Media>
</Layout>
<Layout Class="Parameter" ID="Sheet" Status="Available">
  <Media Dimension="72 72" PartIDKeys="Location">
    <Comment Name="foo">bar</Comment>
    <Media Location="desk"/>
    <Media Location="drawer"/>
  </Media>
</Layout>
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n20020626134204"
      Status="Waiting" JobPartID="ID345" Type="DigitalPrinting" Version="1.4">
  <ResourcePool>
    <!-- Media is partitioned so that it can be referenced
         from the AmountPool
    -->
    <Media Class="Consumable" ID="r0006" PartIDKeys="RunIndex"
          Status="Available">
      <Media RunIndex="0 ~ -1"/>
      <Media RunIndex="1 ~ -2"/>
    </Media>
    <DigitalPrintingParams Class="Parameter" ID="r0007" PartIDKeys="RunIndex"
                           Status="Available">
      <DigitalPrintingParams RunIndex="0 ~ -1">
        <!-- PartAmount with <Part RunIndex="0 ~ -1"/> contains
            the partition details for this MediaRef -->
        <MediaRef rRef="r0006">
          <Part RunIndex="0 ~ -1"/>
        </MediaRef>
      </DigitalPrintingParams>
      <DigitalPrintingParams RunIndex="1 ~ -2">
        <!-- PartAmount with <Part RunIndex="1 ~ -2"/>
            contains the partition details for this MediaRef
      -->
        <MediaRef rRef="r0006">
          <Part RunIndex="1 ~ -2"/>
        </MediaRef>
      </DigitalPrintingParams>
    </DigitalPrintingParams>
    <RunList Class="Parameter" ID="r0008" Status="Available" />
    <Component Class="Quantity" ID="c0008" Status="Unavailable"
               ComponentType="Sheet" />
  </ResourcePool>
  <ResourceLinkPool>
    <MediaLink Usage="Input" rRef="r0006">
      <!-- the AmountPool contains the ResourceLink partition details -->
      <AmountPool>
        <PartAmount Orientation="Flip180">

```

```

        <Part RunIndex="0 -1"/>
    </PartAmount>
    <PartAmount Orientation="Rotate0">
        <Part RunIndex="1 ~ -2"/>
    </PartAmount>
</AmountPool>
</MediaLink>
<DigitalPrintingParamsLink Usage="Input" rRef="r0007"/>
<RunListLink Usage="Input" rRef="r0008"/>
<ComponentLink Usage="Output" rRef="c0008"/>
</ResourceLinkPool>
</JDF>

<LayoutElement Class="Parameter" ID="ID1" PartIDKeys="PageNumber"
    Status="Available">
    <SeparationSpec Name="Cyan"/>
    <SeparationSpec Name="Magenta"/>
    <SeparationSpec Name="Yellow"/>
    <SeparationSpec Name="Black"/>
    <FileSpec/>
    <LayoutElement PageNumber="0"/>
    <LayoutElement PageNumber="1">
        <!--These two SeparationSpec Elements completely replace the
            CMYK in the root
        -->
        <SeparationSpec Name="Black"/>
        <SeparationSpec Name="SpotGreen"/>
    </LayoutElement>
</LayoutElement>
<ExposedMedia Amount="2" Brand="Gooey" Class="Handling" ID="L1"
    PartIDKeys="SheetName Side Separation" Status="Available">
    <Media Dimension="500 600" MediaType="Plate"/>
    <ExposedMedia SheetName="S1">
        <ExposedMedia Side="Front">
            <ExposedMedia ProductID="S1FCPlateJ42" Separation="Cyan"/>
            <ExposedMedia ProductID="S1FMPlateJ42" Separation="Magenta"/>
            <ExposedMedia ProductID="S1FYPlateJ42" Separation="Yellow"
                Status="Unavailable"/>
            <ExposedMedia ProductID="S1FKPlateJ42" Separation="Black"/>
        </ExposedMedia>
        <ExposedMedia Side="Back">
            <ExposedMedia ProductID="S1BCPlateJ42" Separation="Cyan"/>
            <ExposedMedia ProductID="S1BMPlateJ42" Separation="Magenta"/>
            <ExposedMedia ProductID="S1BYPlateJ42" Separation="Yellow"/>
            <ExposedMedia ProductID="S1BKPlateJ42" Separation="Black"/>
        </ExposedMedia>
    </ExposedMedia>
    <ExposedMedia SheetName="S2">
        <ExposedMedia Side="Front">
            <ExposedMedia ProductID="S2FCPlateJ42" Separation="Cyan"/>
            <ExposedMedia ProductID="S2FMPlateJ42" Separation="Magenta"/>
            <ExposedMedia ProductID="S2FYPlateJ42" Separation="Yellow"/>
            <ExposedMedia ProductID="S2FKPlateJ42" Separation="Black"/>
        </ExposedMedia>
    </ExposedMedia>

```

```

        </ExposedMedia>
    </ExposedMedia>
<Preview Class="Parameter" ID="P1" PartIDKeys="PreviewType Separation"
    URL="File:///aaa.pdf" Status="Available">
    <Preview PreviewType="Separation">
        <Preview Separation="Cyan"/>
        <Preview Separation="Magenta"/>
    </Preview>
    <Preview PreviewType="ThumbNail"/>
</Preview>
<Preview Class="Parameter" ID="P2" PartIDKeys="PreviewType Separation"
    Status="Available">
    <Preview Separation="Cyan"/>
    <Preview Separation="Magenta"/>
</Preview>
<Preview Class="Parameter" ID="P3" PartIDKeys="PreviewType Separation"
    URL="File:///aaa.pdf" Status="Available">
    <Preview PreviewType="Separation">
        <Preview Separation="Cyan"/>
    </Preview>
</Preview>
<Preview Class="Parameter" ID="P4" PartIDKeys="PreviewType Separation"
    URL="File:///aaa.pdf" Status="Available">
    <Preview PreviewType="Separation" Separation="Cyan"/>
</Preview>
<Component Class="Quantity" ID="c1" PartIDKeys="SheetName" Status="Available"
    ComponentType="Sheet">
    <Component SheetName="Sheet 1"/>
</Component>
<Component Class="Quantity" ID="c2" PartIDKeys="SheetName" Status="Available"
    ComponentType="Sheet">
    <Component SheetName="Sheet 2"/>
</Component>
<FoldingParams Class="Parameter" ID="fold" NoOp="true" PartIDKeys="SheetName"
    Status="Available">
    <FoldingParams NoOp="false" SheetName="Sheet 1"/>
</FoldingParams>
<Component Class="Quantity" ID="c12" PartIDKeys="SheetName" SheetName="Sheet 1"
    Status="Available" ComponentType="Sheet"/>
<Component Class="Quantity" ID="c22" PartIDKeys="SheetName" SheetName="Sheet 2"
    Status="Available" ComponentType="Sheet"/>
<FoldingParams Class="Parameter" ID="fold2" NoOp="true" PartIDKeys="SheetName"
    Status="Available">
    <FoldingParams NoOp="false" />
</FoldingParams>
<ExposedMedia Class="Handling" ID="L1" PartIDKeys="Separation"
    Status="Available">
    <Media Brand="foo" MediaType="Film"/>
    <ExposedMedia Separation="Cyan"/>
    <ExposedMedia Separation="Magenta">
        <Media Brand="bar" MediaType="Film"/>
    </ExposedMedia>
</ExposedMedia>
<ExposedMedia Class="Handling" ID="L21" PartIDKeys="Separation"
    Status="Available">
    <Media MediaType="Film"/>
    <ExposedMedia Separation="Cyan">
        <Media Brand="foo"/>
    </ExposedMedia>
    <ExposedMedia Separation="Magenta">

```

```

<Media Brand="bar" Class="Consumable"/>
</ExposedMedia>
</ExposedMedia>
<ExposedMedia Class="Handling" ID="L31" PartIDKeys="Separation" Status="Available">
    <Media MediaType="Film">
        <Media Brand="foo" Separation="Cyan"/>
        <Media Brand="bar" Separation="Magenta"/>
    </Media>
</ExposedMedia>
<Media Class="Consumable" ID="MediaID" MediaType="Film" PartIDKeys="Separation"
      Status="Available">
    <Media Brand="foo" Separation="Cyan"/>
    <Media Brand="bar" Separation="Magenta"/>
</Media>
<ExposedMedia Class="Handling" ID="L41" PartIDKeys="Separation"
      Status="Available">
    <ExposedMedia Separation="Cyan">
        <!--equivalent to <Media MediaType="Film" Brand="foo"/> -->
        <MediaRef rRef="MediaID">
            <Part Separation="Cyan"/>
        </MediaRef>
    </ExposedMedia>
    <ExposedMedia Separation="Magenta">
        <!--equivalent to <Media MediaType="Film" Brand="bar"/> -->
        <MediaRef rRef="MediaID">
            <Part Separation="Magenta"/>
        </MediaRef>
    </ExposedMedia>
</ExposedMedia>
<Media Class="Consumable" ID="MediaID2" MediaType="Film"
      PartIDKeys="Separation" Status="Available">
    <Media Brand="foo" Separation="Cyan"/>
    <Media Brand="bar" Separation="Magenta"/>
</Media>
<ExposedMedia Class="Handling" ID="L51" PartIDKeys="Separation"
      Status="Available">
    <MediaRef rRef="MediaID2"/>
</ExposedMedia>
<ExposedMedia Class="Handling" ID="L1" PartIDKeys="SheetName Side Separation"
      Status="Available">
    <Media Class="Consumable" MediaType="Film"/>
    <ExposedMedia SheetName="S1">
        <ExposedMedia Side="Front">
            <ExposedMedia ProductID="1" Separation="Cyan"/>
            <ExposedMedia ProductID="2" Separation="Magenta"/>
            <ExposedMedia ProductID="3" Separation="Yellow"/>
            <ExposedMedia ProductID="4" Separation="Black"/>
        </ExposedMedia>
        <!-- Master partition that is referenced by an Identical Element -->
        <ExposedMedia Side="Back">
            <ExposedMedia ProductID="5" Separation="Cyan"/>
            <ExposedMedia ProductID="6" Separation="Magenta"/>
            <ExposedMedia ProductID="7" Separation="Yellow"/>
            <ExposedMedia ProductID="8" Separation="Black"/>
        </ExposedMedia>
    </ExposedMedia>
    <ExposedMedia SheetName="S2">
        <ExposedMedia Side="Front">
            <ExposedMedia ProductID="9" Separation="Cyan"/>

```

```

        <ExposedMedia ProductID="10" Separation="Magenta"/>
        <ExposedMedia ProductID="11" Separation="Yellow"/>
        <ExposedMedia ProductID="12" Separation="Black"/>
    </ExposedMedia>
    <!-- Logical partition with an Identical Element -->
    <ExposedMedia Side="Back">
        <Identical>
            <Part SheetName="S1" Side="Back"/>
        </Identical>
    </ExposedMedia>
</ExposedMedia>
<ExposedMediaLink Usage="Input" rRef="L1">
    <Part SheetName="S2" Side="Back" Separation="Black"/>
</ExposedMediaLink>
<ExposedMedia Class="Handling" ID="L2" PartIDKeys="SheetName Side Separation"
    Status="Available">
    <ExposedMedia SheetName="S1">
        <ExposedMedia Side="Front">
            <ExposedMedia ProductID="1" Separation="Cyan"/>
            <ExposedMedia ProductID="2" Separation="Magenta"/>
            <ExposedMedia ProductID="3" Separation="Yellow"/>
            <ExposedMedia ProductID="4" Separation="Black"/>
        </ExposedMedia>
        <ExposedMedia Side="Back">
            <ExposedMedia ProductID="5" Separation="Cyan"/>
            <ExposedMedia ProductID="6" Separation="Magenta"/>
            <ExposedMedia ProductID="7" Separation="Yellow"/>
            <ExposedMedia ProductID="8" Separation="Black"/>
        </ExposedMedia>
    </ExposedMedia>
    <ExposedMedia SheetName="S2">
        <ExposedMedia Side="Front">
            <ExposedMedia ProductID="9" Separation="Cyan">
                <!--This Identical is invalid because it references from a Separation
                    partition to a Surface partition -->
                <Identical>
                    <Part SheetName="S1" Side="Back"/>
                </Identical>
            </ExposedMedia>
        </ExposedMedia>
    </ExposedMedia>
</ExposedMedia>

<ResourcePool>
    <ExposedMedia Class="Handling" ID="L1" PartIDKeys="Location"
        Status="Available">
        <ExposedMedia Amount="42" Location="dd1">
            <Location LocID="PP_01234" LocationName="Desk Drawer 1"/>
        </ExposedMedia>
        <ExposedMedia Amount="100" Location="dd2">
            <Location LocID="PP_01235" LocationName="Desk Drawer 2"/>
        </ExposedMedia>
        <Media/>
    </ExposedMedia>
</ResourcePool>
<ResourceLinkPool>

```

```

<ExposedMediaLink Amount="50" Usage="Input" rRef="L1">
    <Part Location="dd2"/>
    <!-- Note that @Location can but need not match
        Location/@LocationName
    -->
    </ExposedMediaLink>
</ResourceLinkPool>
<Media Class="Consumable" ID="TopMedia" Status="Available">
    <Location LocationName="Top"/>
</Media>
<Media Class="Consumable" ID="BottomMedia" Status="Available">
    <Location LocationName="Bottom"/>
</Media>
<LayoutPreparationParams Class="Parameter" ID="L1" PartIDKeys="RunIndex"
    Sides="TwoSidedFlipY" Status="Available">
    <!-- Partition that defines the first and last page of the document -->
    <LayoutPreparationParams RunIndex="0 1 -2 -1">
        <MediaRef rRef="TopMedia"/>
    </LayoutPreparationParams>
    <!-- Partition that defines the inside pages of the document -->
    <LayoutPreparationParams RunIndex="2 ~ -3">
        <MediaRef rRef="BottomMedia"/>
    </LayoutPreparationParams>
</LayoutPreparationParams>

<ResourcePool>
    <RunList Class="Parameter" ID="SheetSurfacesGeneratedByImposition"
        PartIDKeys="SheetIndex" Status="Available">
        <RunList SheetIndex="1"/>
        <RunList SheetIndex="3"/>
        <RunList SheetIndex="5"/>
    </RunList>
</ResourcePool>
<ResourceLinkPool>
    <RunListLink rRef="SheetSurfacesGeneratedByImposition" Usage="Output">
        <!-- output of imposition -->
        <Part SheetIndex="5"/>
        <Part SheetIndex="1"/>
    </RunListLink>
</ResourceLinkPool>

<ExposedMediaLink Usage="Input" rRef="E1">
    <Part Separation="Cyan" SheetName="S1"/>
    <Part Separation="Magenta" SheetName="S1"/>
    <AmountPool>
        <PartAmount>
            <Part Separation="Cyan" SheetName="S1" Side="Front"/>
        </PartAmount>
        <PartAmount>
            <Part Separation="Cyan" SheetName="S1" Side="Back"/>
        </PartAmount>
        <PartAmount>
            <Part Separation="Magenta" SheetName="S1" Side="Front"/>
        </PartAmount>
        <PartAmount Amount="2">
            <Part Separation="Magenta" SheetName="S1" Side="Back"/>
        </PartAmount>
    </AmountPool>
</ExposedMediaLink>

```

SHALL be<**ResourceLinkPool**>

```

<ExposedMediaLink Usage="Input" rRef="XM_ID">
    <Part SheetName="S1" Side="Front" Separation="Black" PartVersion="Deutsch"/>
</ExposedMediaLink>
</ResourceLinkPool>
<ResourcePool>
    <ExposedMedia Brand="Gooey" Class="Handling" ID="XM_ID"
        PartIDKeys="SheetName Side Separation PartVersion"
        PartUsage="Implicit" ProductID="Root" Status="Available">
        <Media Dimension="500 600" MediaType="Plate"/>
        <ExposedMedia ProductID="S1" SheetName="S1">
            <ExposedMedia ProductID="S1F" Side="Front">
                <ExposedMedia ProductID="S1FC" Separation="Cyan"/>
                <ExposedMedia ProductID="S1FM" Separation="Magenta"/>
                <ExposedMedia ProductID="S1FY" Separation="Yellow"/>
                <ExposedMedia ProductID="S1FK" Separation="Black">
                    <ExposedMedia ProductID="S1FKD" PartVersion="Deutsch"/>
                    <ExposedMedia ProductID="S1FKE" PartVersion="English"/>
                </ExposedMedia>
            </ExposedMedia>
            <ExposedMedia ProductID="S1B" Side="Back">
                <ExposedMedia ProductID="S1BC" Separation="Cyan"/>
                <ExposedMedia ProductID="S1BM" Separation="Magenta"/>
                <ExposedMedia ProductID="S1BY" Separation="Yellow"/>
                <ExposedMedia ProductID="S1BK" Separation="Black">
                    <ExposedMedia ProductID="S1BKD" PartVersion="Deutsch"/>
                    <ExposedMedia ProductID="S1BKE" PartVersion="English"/>
                </ExposedMedia>
            </ExposedMedia>
        </ExposedMedia>
        <ExposedMedia ProductID="S2" SheetName="S2">
            <ExposedMedia ProductID="S2F" Side="Front">
                <ExposedMedia ProductID="S2FC" Separation="Cyan"/>
                <ExposedMedia ProductID="S2FM" Separation="Magenta"/>
                <ExposedMedia ProductID="S2FY" Separation="Yellow"/>
                <ExposedMedia ProductID="S2FK" Separation="Black"/>
            </ExposedMedia>
            <ExposedMedia Side="Back">
                <Identical>
                    <Part SheetName="S1" Side="Back"/>
                </Identical>
            </ExposedMedia>
        </ExposedMedia>
    </ResourcePool>
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="Link0037" Status="Waiting"
    Type="Gathering" JobPartID="ID345" Version="1.4" >
    <ResourcePool>
        <GatheringParams Class="Parameter" ID="Gath01" Locked="false"
            Status="Available"/>
        <Component Class="Quantity" ComponentType="Sheet"
            DescriptiveName="printed insert sheets" ID="Sheets01"
            PartIDKeys="SheetName" Status="Available">
            <Component SheetName="Sheet1"/>
            <Component SheetName="Sheet2"/>
            <Component SheetName="Sheet3"/>
        </Component>
    </ResourcePool>

```

```

<Component Class="Quantity" ComponentType="Sheet"
           ID="SheetsOut" Status="Available"/>
</ResourcePool>
<ResourceLinkPool>
    <GatheringParamsLink Usage="Input" rRef="Gath01"/>
    <!--three ComponentLink explicitly reference individual parts -->
    <ComponentLink Usage="Input" rRef="Sheets01">
        <Part SheetName="Sheet1"/>
    </ComponentLink>
    <ComponentLink Usage="Input" rRef="Sheets01">
        <Part SheetName="Sheet2"/>
    </ComponentLink>
    <ComponentLink Usage="Input" rRef="Sheets01">
        <Part SheetName="Sheet3"/>
    </ComponentLink>
    <ComponentLink Usage="Output" rRef="SheetsOut"/>
</ResourceLinkPool>
</JDF>
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="Link0037" Status="Waiting"
      Type="Gathering" JobPartID="ID345" Version="1.4">
    <ResourcePool>
        <GatheringParams Class="Parameter" ID="Gath01" Locked="false"
                         Status="Available"/>
        <Component Class="Quantity" ComponentType="Sheet"
                  DescriptiveName="printed insert sheets" ID="Sheets01"
                  PartIDKeys="SheetName" Status="Available">
            <Component SheetName="Sheet1"/>
            <Component SheetName="Sheet2"/>
            <Component SheetName="Sheet3"/>
        </Component>
        <Component Class="Quantity" ComponentType="Sheet"
                  ID="SheetsOut" Status="Available"/>
    </ResourcePool>
    <ResourceLinkPool>
        <GatheringParamsLink Usage="Input" rRef="Gath01"/>
        <!--the ComponentLink implicitly references all three parts -->
        <ComponentLink Usage="Input" rRef="Sheets01"/>
        <ComponentLink Usage="Output" rRef="SheetsOut"/>
    </ResourceLinkPool>
</JDF>

```

3.7 AuditPool and Audit

Audit Elements contain the post-facto recorded results of a Process such as the execution of a JDF Node or modification of the JDF itself. Audit Elements become static after a Process has been finished. They SHALL NOT be modified after the Process has been aborted or completed. Therefore, if AuditStatus or ResourceAudit^{%%%Audit} Audit Elements link to Resources, those Resources SHOULD be locked in order to inhibit accidental modification of audited information, which is why XJDF includes a locking mechanism for Resources. Audit Elements record any event related to the following situations:

- The creation of a JDF Node by a Created Audit.
- General information by a Notification %%%Audit.
- Actual data about the production and Resource consumption by a ResourceAudit %%%Audit.
- Any Process phase times. Examples include setting up a Device, maintenance and washing, as well as down-times as a result of failure, breaks or pauses.

- Actual final Process execution data. For example, the Process start and end times, as well as the final Process state, as determined by a `ProcessRun` Audit.

Audit information might be used by MIS for operations such as evaluation or invoicing. The Figure 3-5 depicts the structure of the `AuditPool` and `Audit` Elements, such as `Created`. Audit entries are ordered chronologically, with the last entry in the `AuditPool` representing the newest. A `ProcessRun` Element containing the scheduling data finalizes each Process run. All subsequent entries belong to the next run.

3.7.1 AuditPool

The following table defines the contents of the `AuditPool` Element.

Table 3-11: AuditPool Element

Name	Data Type	Description
<code>AuditStatus</code> *	element	Chronologically ordered list of any of the elements specified in this table. See Table 3-12, “List of Audit Elements” on page 81
<code>Created</code> *	element	Chronologically ordered list of any of the elements specified in this table. See Table 3-12, “List of Audit Elements” on page 81
<code>Notification</code> * *%%	element	Chronologically ordered list of any of the elements specified in this table. See Table 3-12, “List of Audit Elements” on page 81
<code>ProcessRun</code> *	element	Chronologically ordered list of any of the elements specified in this table. See Table 3-12, “List of Audit Elements” on page 81
<code>ResourceAudit</code> *%%	element	Chronologically ordered list of any of the elements specified in this table. See Table 3-12, “List of Audit Elements” on page 81

3.7.2 Audit

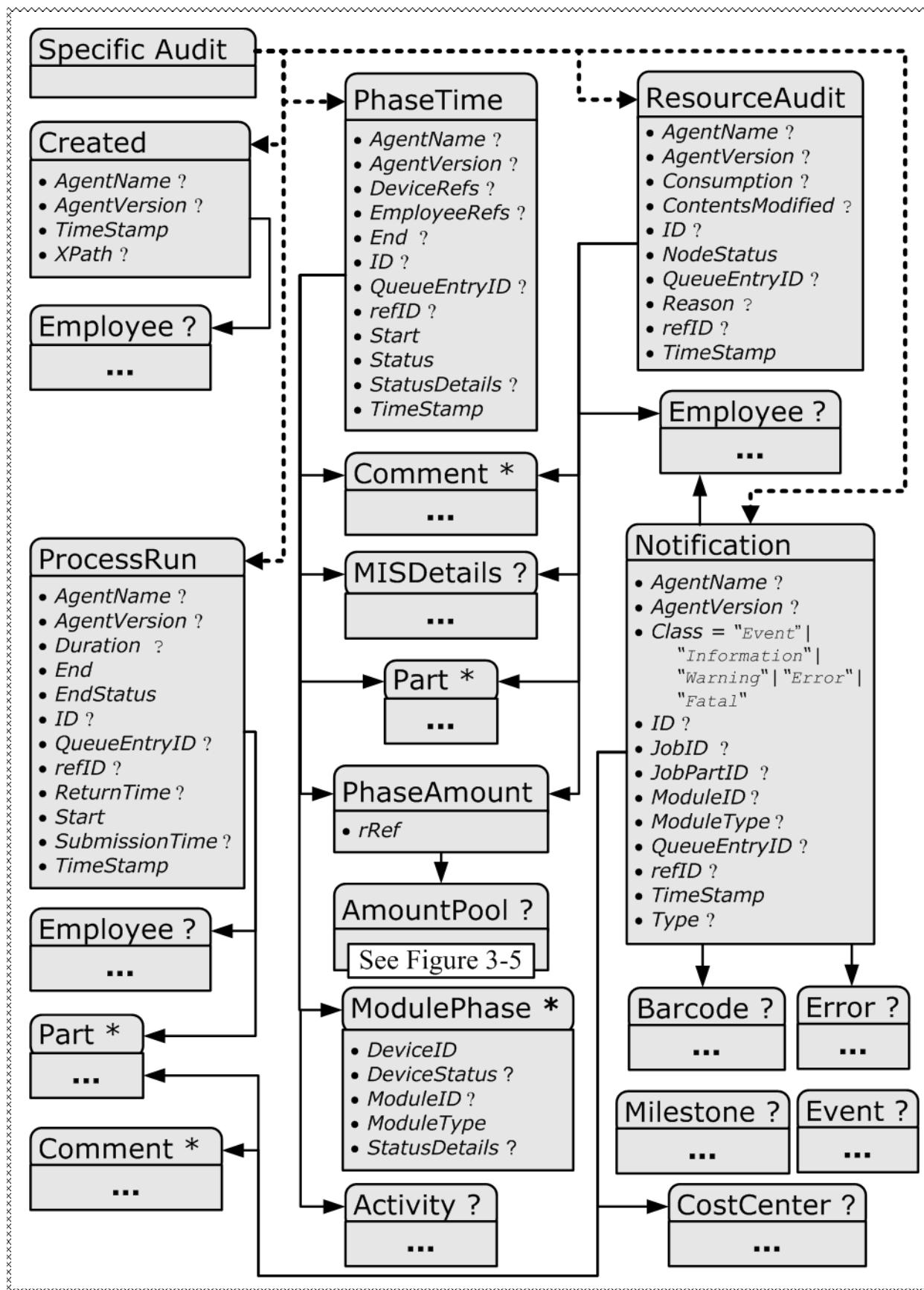
Table 3-12: List of Audit Elements

Name	Page	Description
<code>AuditNotification</code>	page 159 %%	Logs individual events that occurred during processing
<code>AuditResource</code>	page 169 %%	Describes the usage of Resources during execution of a Node or the modification of the intended usage of a Resource
<code>AuditStatus</code>	page 163	Logs start and end times of any Process states and substates, denoted as phases. Phases can reflect any arbitrary subdivisions of a Process.
<code>Created</code>	page 84	Logs creation of JDF Node or Resource
<code>ProcessRun</code>	page 85	Summarizes one complete execution run of a Node or delimits a group of Audit Elements for each individual Process run.

3.7.2.1 Structure Diagram of Audit Elements

Figure 3-5 shows the structure of the `Audit`. Arrows point to child elements. Arrows with dashed lines point to each Audit.

Figure 3-5: Specific Audit – a Diagram of its Structure



3.7.2.2 Created

This Element allows the creation of a JDF Node or Resource to be logged.

Table 3-13: Created Audit Element

Name	Data Type	Description
<i>AgentName</i> ?	string	The name of the application that added the <i>Created</i> Element to the <i>AuditPool</i> (and was responsible for the creation or modification). Both the company name and the product name MAY appear, and SHOULD be consistent between versions of the application.
<i>AgentVersion</i> ?	string	The version of the application that added the <i>Created</i> Element to the <i>AuditPool</i> (and was responsible for the creation or modification). The format of the version string MAY vary from one application to another, but SHOULD be consistent for an individual application.
<i>ContactRef</i> ?	IDREF	Employee who created this <i>Created</i> Element.
<i>TimeStamp</i>	dateTime	Records the date and time when the Element referred to by this <i>Created</i> Audit was created.
<i>XPath</i> ?	XPath	@ <i>XPath</i> SHALL refer to the created Elements or Attributes.

3.7.2.3 AuditNotification

AuditNotification contains information about individual events that occurred during processing. For a detailed discussion of event properties, see Section 4.7, “Error Handling”.

AuditNotification is syntactically the same as *SignalNotification*. A device SHOULD write an *AuditNotification* element for every *SignalNotification* XJMF that it emits.

Table 3-14: AuditNotification Element

Name	Data Type	Description
<i>Author</i> ?	string	See <i>Message/@Author</i> .
<i>DeviceID</i> ?	NMTOKEN	See <i>Message/@DeviceID</i> .
<i>ID</i>	ID	See <i>Message/@ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>refID</i> ?	NMTOKEN	See <i>Signal/@refID</i> .
<i>Time</i> ?	dateTime	See <i>Message/@Time</i> .
<i>Notification</i> ?	element	Notification that describes the event. See Section 5.5.6.3.1, “Notification” on page 227.

3.7.2.4 AuditStatus

AuditStatus contains audit information about the start and end times of any Process states and substates, denoted as phases. Phases can reflect any arbitrary subdivisions of a Process, such as maintenance, washing, plate changing, failures and breaks. *AuditStatus* Elements SHOULD be written for every significant status change that is detected. *AuditStatus* is syntactically the same as *SignalStatus*. Whereas XJMF/*SignalStatus* conveys the momentary status of a device and or job, *AuditStatus* conveys the status during an entire phase. A device SHALL not write new *AuditStatus* elements for every *SignalStatus* XJMF that it emits.

AuditStatus Elements MAY also be used to log the actual time spans when *Resources* are used by a Process. For example, the temporary usage of a fork lift can be logged if an *AuditStatus* Element is added that contains an

`AuditStatus/DeviceInfo/@DeviceID` of the fork lift and specifies the actual start and end time of the usage of that fork lift.

Note: AuditStatus is syntactically the same as SignalStatus.

Table 3-15: AuditStatus Element

Name	Data Type	Description
<code>Author?</code>	string	See <code>Message/@Author</code> .
<code>DeviceID?</code>	NMTOKEN	See <code>Message/@DeviceID</code> .
<code>ID?</code>	ID	See <code>Message/@ID</code> .
<code>PersonalID?</code>	NMTOKEN	See <code>Message/@PersonalID</code> .
<code>refID?</code>	NMTOKEN	See <code>Signal/@refID</code> .
<code>Time?</code>	dateTime	See <code>Message/@Time</code> .
<code>DeviceInfo</code> + 0/0/0 0/0/0	element	Describes the actual device Status. If multiple <code>DeviceInfo</code> Elements are specified, these describe multiple Devices. A sequential state change of an individual Device SHALL be encoded as 2 separate Signals.

3.7.2.5 ProcessRun

ProcessRun summarizes one execution of a process. A ProcessRun element SHALL be written each time an XJDF is returned to a Controller.

All Job related Amounts in subsequent Audit elements and XJMF messages SHALL restart at 0 when a Node is processed on a device after a ProcessRun has been sent.

Table 3-16: ProcessRun Audit Element (Sheet 1 of 2)

Name	Data Type	Description
<code>AgentName?</code>	string	The name of the application that added the <code>PROCESSRUN</code> Element to the <code>AuditPool</code> (and was responsible for the creation or modification). Both the company name and the product name MAY appear, and SHOULD be consistent between versions of the application.
<code>AgentVersion?</code>	string	The version of the application that added the <code>ProcessRun</code> Element to the <code>AuditPool</code> (and was responsible for the creation or modification). The format of the version string MAY vary from one application to another, but SHOULD be consistent for an individual application.
<code>Duration?</code>	duration	Time span of the effective Process runtime without intentional or unintentional breaks. That time span is the sum of all Process phases when the <code>NodeInfo/@NodeStatus</code> is " <code>InProgress</code> ", " <code>Setup</code> " or " <code>Cleanup</code> ".
<code>End</code>	dateTime	Date and time at which the Process ended.

Table 3-16: ProcessRun Audit Element (Sheet 2 of 2)

Name	Data Type	Description
<i>EndStatus</i>	enumeration	The <i>NodeInfo/@NodeStatus</i> of the Process at the end of the run. For a description of Process states, see Table 8-148, “NodeStatus Attribute Values” on page 710. Allowed values are: <i>Aborted</i> – The Node has been aborted before producing the desired result. <i>Completed</i> – The Node has been completed and the desired result has been produced.
<i>ID?</i>	ID	@ <i>ID</i> of the <i>ProcessRun</i> . @ <i>ID</i> SHALL be specified if there is support to subsequently create correction <i>ProcessRun</i> Elements.
<i>QueueEntryID?</i>	NMTOKEN	@ <i>QueueEntryID</i> of the <i>QueueEntry</i> during which this <i>ProcessRun</i> was generated.
<i>refID?</i>	IDREF	Reference to a previous <i>ProcessRun</i> that this <i>ProcessRun</i> corrects. The referenced <i>ProcessRun</i> SHALL reside in the same <i>AuditPool</i> .
<i>ReturnTime?</i>	dateTime	Date and Time of the <i>ReturnQueueEntry</i> submission.
<i>Start</i>	dateTime	Date and time at which the Process started.
<i>SubmissionTime?</i>	dateTime	Date and Time of the <i>SubmitQueueEntry</i> submission. This value SHOULD be identical with <i>QueueEntry/@SubmissionTime</i> .
<i>TimeStamp</i>	dateTime	Records the date and time when this <i>ProcessRun</i> Audit was appended to the <i>AuditPool</i> .
<i>Contact?</i>	element	Employee who created this <i>ProcessRun</i> Element.
<i>Part</i> *	element	Describes which parts of a Process this <i>ProcessRun</i> belongs to. If <i>Part</i> is not specified for a <i>ProcessRun</i> , it refers to all parts. For example, imagine a print Job that is to produce three different Sheets. All Sheets are described by one Partitioned Resource. The <i>Part</i> Elements define, unambiguously, the processing of the Sheet to which the <i>ProcessRun</i> refers.

3.7.2.6 AuditResource

The *AuditResource* Element describes the usage of Resources during execution of a process). It logs consumption and production amounts of any quantifiable Resources, accumulated over one Process run or one part of a Process run. *AuditResource* is syntactically the same as *SignalResource*. Whereas XJMF/*SignalResource* MAY convey the momentary consumption or production of a Resource, *AuditResource* conveys the consumption or production of a Resource during an entire phase. A device SHALL write a copy of the last *SignalResource* that it emits during a *ProcessRun* as an *AuditResource*.

Table 3-17: AuditResource Element (Sheet 1 of 2)

Name	Data type	Description
<i>Author?</i>	string	See <i>Message/@Author</i> .
<i>DeviceID?</i>	NMTOKEN	See <i>Message/@DeviceID</i> .
<i>ID</i>	ID	See <i>Message/@ID</i> .
<i>PersonalID?</i>	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>refID?</i>	NMTOKEN	See <i>Signal/@refID</i> .
<i>Time?</i>	dateTime	See <i>Message/@Time</i> .

Table 3-17: AuditResource Element (Sheet 2 of 2)

Name	Data Type	Description
ResourceInfo *	element	Signal from a Command: contains information about the Resources after modification. Signal from a Query: contains the amount data of Resources and if requested, the Resources itself.

1

Example 3-4: ResourceAudit: Before Logging

The following example describes the logging of a modification of the media weight and amount. The XJDF document before modification requests 400 copies of 80 gram media.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="J1" Status="Waiting"
      Type="ConventionalPrinting" JobPartID="ID234" Version="1.4">
  <ResourceLinkPool>
    <MediaLink Amount="400" Usage="Input" rRef="RLink"/>
    <ConventionalPrintingParamsLink Usage="Input" rRef="R01"/>
    <ComponentLink Usage="Output" rRef="R02"/>
  </ResourceLinkPool>
  <ResourcePool>
    <Media ID="RLink" Class="Consumable" Status="Available"
          Amount="400" Weight="80"/>
    <ConventionalPrintingParams ID="R01" Class="Parameter" Status="Available"/>
    <Component ID="R02" Class="Quantity" Status="Unavailable"
               ComponentType="Sheet"/>
  </ResourcePool>
</JDF>
```

Example 3-5: ResourceAudit: Logging of Consumption

The XJDF after modification specifies that 421 copies of 90-gram media have been consumed.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="J1" Status="Waiting"
      Type="ConventionalPrinting" JobPartID="ID234" Version="1.4">
  <ResourceLinkPool>
    <!-- Note that ActualAmount has been added to the ResourceLink -->
    <MediaLink ActualAmount="421" Amount="400" Usage="Input" rRef="RLink"/>
    <ConventionalPrintingParamsLink Usage="Input" rRef="R01"/>
    <ComponentLink Usage="Output" rRef="R02"/>
  </ResourceLinkPool>
  <ResourcePool>
    <Media ID="RPrev" Class="Consumable" Status="Available" Amount="400"
          Weight="80"/>
    <!--Copy of the original resource-->
    <Media ID="RLink" Class="Consumable" Status="Available" Amount="421"
          Weight="90"/>
    <ConventionalPrintingParams ID="R01" Class="Parameter" Status="Available"/>
    <Component ID="R02" Class="Quantity" Status="Unavailable"
               ComponentType="Sheet"/>
    <!--modified resource-->
  </ResourcePool>
  <AuditPool>
    <ResourceAuditTimeStamp="2008-08-28T18:20:00Z">
      <MediaLink ActualAmount="421" Amount="400" Usage="Input" rRef="RLink"/>
      <MediaLink Amount="400" Usage="Input" rRef="RPrev"/>
    </ResourceAudit>
  </AuditPool>
</JDF>
```

```

</AuditPool>
</JDF>

<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="J1" Status="Waiting"
      Type="Product" JobPartID="ID234" Version="1.4">
  <ResourceLinkPool>
    <MediaIntentLink Usage="Input" rRef="id">
      <Part Option="Option2"/>
    </MediaIntentLink>
    <ComponentLink Usage="Output" rRef="R02"/>
  </ResourceLinkPool>
  <ResourcePool>
    <MediaIntent ID="id" PartIDKeys="Option">
      <!-- the common MediaIntent resource details -->
      <MediaIntent Option="Option1">
        <Weight Preferred="80" DataType="NumberSpan"/>
      </MediaIntent>
      <MediaIntent Option="Option2">
        <Weight Preferred="90" DataType="NumberSpan"/>
      </MediaIntent>
    </MediaIntent>
    <Component ID="R02" Class="Quantity" Status="Unavailable"
               ComponentType="Sheet"/>
  </ResourcePool>
  <AuditPool>
    <ResourceAudit>
      <!-- the actual MediaIntent ResourceLink -->
      <MediaIntentLink Usage="Input" rRef="id">
        <Part Option="Option2"/>
      </MediaIntentLink>
      <!-- the original MediaIntent ResourceLink -->
      <MediaIntentLink Usage="Input" rRef="id">
        <Part Option="Option1"/>
      </MediaIntentLink>
    </ResourceAudit>
  </AuditPool>
</JDF>

```

3.8 XJDF Extensibility

The XJDF specification aims to support plug-and-play as much as possible. None the less, **XJDF** is meant to be flexible and therefore useful to any vendor, as each vendor MAY have specific data to include in the **XJDF** files. However, foreign namespace extensions SHOULD NOT duplicate functionality of XJDF-defined Attributes and Elements. This section describes how **XJDF** MAY be extended.

Namespaces in XML XJDF Extensibility is implemented using XML Namespaces [XMLNS].

Example 3-6: Namespaces in XML

The example illustrates how private namespaces are declared and used to extend an existing **XJDF** Resource by adding private Attributes and a private Element.

```

<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1"
      xmlns:foo="fooschema URI" ID="ID1" Status="Ready"
      JobPartID="ID345" Version="1.4" >
  <!-- ... -->
  <SomeJDFDefinedResource name="abc" foo:specialname="cba">
  <!-- ... -->
  <foo:PrivateStuff type="" />
  <!-- ... -->
</SomeJDFDefinedResource>

```

```
<!-- ... -->
</JDF>
```

3.8.0.1 XJDF Namespace

The official namespace URI for XJDF Version 2.0 is: http://www.CIP4.org/XJDFSchema_2_0. It is strongly RECOMMENDED to use either the default namespace with no prefix or a prefix of “jdf” as the XJDF namespace prefix.

3.8.1 Foreign Namespaces

Attributes in a foreign namespace MAY be added to any XJDF element. Elements in a foreign namespace SHALL NOT be specified in JDF elements other than Notification, Resource and Product Intent elements.

3.8.2 Creating Extension Assets:

New Assets may be defined by creating a **ResourceSet** with **@Name** referring to a proprietary xml namespace. The Extension Element SHALL reside in the appropriate child **Resource** element.

Example 3-7: Creating Extension Assets

```
<ResourceSet Name="foo:bar" xmlns:foo="www.foo.com">
  <Resource>
    <foo:bar attrib="myAttrib"/>
  </Resource>
</ResourceSet>
```

3.8.3 Extending Process Types

XJDF defines a basic set of Process types. However, because XJDF allows flexible encoding, this list, by definition, will not be complete. Vendors that have specific Processes that do not fit in the general XJDF Processes and that are not combinations of individual XJDF Processes (see Section 3.3.3, “Process Nodes”) can create XJDF Process Nodes of their own type. Then the content of the **@Types** Attribute MAY be specified with a prefix that identifies the organization. The prefix and name SHALL be separated by a single colon (:) as shown in the following example.

Example 3-8: Extending Process Types

```
<JDF Type="myCompaniesNS:MyVeryImportantProcess"
      xmlns="http://www.CIP4.org/JDFSchema_1_1"
      xmlns:myCompaniesNS="my companies namespace URI"
      ID="ID1" JobPartID="ID345" Status="Ready" Version="1.4" >
  <!-- ... -->
</JDF>
```

3.8.3.1 Rules about Process Extension

The use of namespace prefixes in the **@Types** Attribute is for extensions only. Standard XJDF Process types SHALL be specified without a prefix in XJDF/**@Types**. If a Process is simply an extension of an existing Process, it is possible to describe the private data by extending the existing Resource types. This is described in greater detail in the sections below.

3.8.4 Extending NMOKEN Lists

Many Resources contain Attributes of type NMOKEN and some of these have a set of predefined, suggested enumerative values. These lists MAY be extended with private keywords. In order to identify private keywords, it is strongly RECOMMENDED to prefix these keywords with a namespace-like syntax (i.e., a namespace prefix separated by a single colon “:”). Such a namespace prefix SHOULD be defined in the XJDF ticket with the standard xmlns:Prefix=“someURI” notation, even if no extension Elements or Attributes from that namespace occur in the XJDF ticket. Implementations that find an unknown NMOKEN prefixed by a namespace prefix MAY then attempt to use the default value of that Attribute.

Example 3-9: Extending NMOKEN Lists

For instance, if an implementation encounters **TrappingParams/@TrapEndStyle** (see below in Table 3-18) in the **XJDF** snippet shown below, and if the implementation does not support the "*HDM*" extension, the best assumption is to use **@TrapEndStyle = "Miter"**, which is the default for **@TrapEndStyle**.

```
<TrappingParams TrapEndStyle="HDM:FooBar"/>
```

Table 3-18: Excerpt from TrappingParams

Name	Data Type	Description
<i>TrapEndStyle</i> ?	NMOKEN	<p>Instructs the trap engine how to form the end of a trap that touches another object.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Miter</i> <i>Overlap</i> <p>Note: other values might be added later as a result of customer requests.</p>

3.8.5 Future XJDF Extensions

In future versions, certain private extensions will become more widely used, even by different vendors. As private extensions become more of a general rule, those extensions will be candidates for inclusion in the next version of the **XJDF** specification. At that time the specific extensions will have to be described and will be included into the **XJDF** namespace.

- **XJDF**

Chapter 4 Life Cycle of XJDF

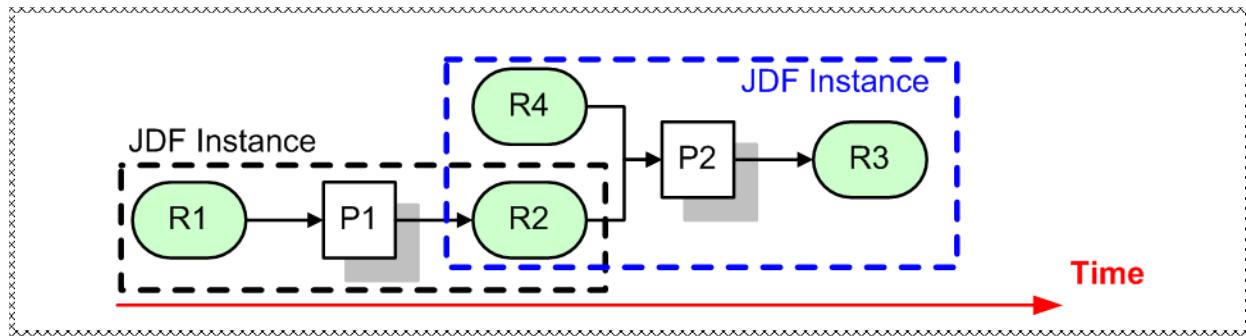
```
JMF<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="Job1" JobID="J1"
    JobPartID="P1"
    Status="Waiting" Type="Product" Version="1.4">
    <ResourcePool>
        <Component Amount="10000" Class="Quantity"
            DescriptiveName="Complete 16-page Brochure" ID="Link0003"
            Status="Unavailable" ComponentType="Sheet" />
        <LayoutIntent Class="Intent" ID="Link0004" Status="Available">
            <Dimensions DataType="XYPairSpan" Preferred="612 792"
                Range="576 720 ~ 648 864"/>
            <Pages DataType="IntegerSpan" Preferred="16"/>
        </LayoutIntent>
        <MediaIntent Class="Intent" ID="Link0005" PartIDKeys="Option"
            Status="Available">
            <FrontCoatings DataType="NameSpan" Preferred="None"/>
            <MediaIntent Option="1">
                <FrontCoatings DataType="NameSpan" Preferred="Glossy"/>
            </MediaIntent>
            <BackCoatings DataType="NameSpan" Preferred="None"/>
        </MediaIntent>
        <ProductionIntent Class="Intent" ID="ID_PI" Status="Available">
            <ScreeningParamsRef rRef="ScreenID"/>
        </ProductionIntent>
        <ScreeningParams Class="Parameter" ID="ScreenID" Status="Incomplete">
            <ScreenSelector ScreeningFamily="My favorite screen"
                SpotFunction="Ellipse"/>
        </ScreeningParams>
    </ResourcePool>
    <ResourceLinkPool>
        <ComponentLink Usage="Output" rRef="Link0003"/>
        <LayoutIntentLink Usage="Input" rRef="Link0004"/>
        <MediaIntentLink Usage="Input" rRef="Link0005"/>
        <ProductionIntentLink Usage="Input" rRef="ID_PI"/>
    </ResourceLinkPool>
</JDF>

<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="Job1" JobID="J1"
    JobPartID="P1"
    Status="Waiting" Type="Product" Version="1.4">
    <ResourcePool>
        <Component Amount="10000" Class="Quantity"
            DescriptiveName="Complete 16-page Brochure" ID="Link0003"
            Status="Unavailable" ComponentType="Sheet" />
        <LayoutIntent Class="Intent" ID="Link0004" Status="Available">
            <Dimensions DataType="XYPairSpan" Preferred="612 792"
                Range="576 648 720 864"/>
            <Pages DataType="IntegerSpan" Preferred="16"/>
        </LayoutIntent>
        <MediaIntent Class="Intent" ID="Link0005" PartIDKeys="Option"
            Status="Available">
            <FrontCoatings DataType="NameSpan" Preferred="None"/>
            <MediaIntent Option="1">
                <FrontCoatings DataType="NameSpan" Preferred="Glossy"/>
            </MediaIntent>
            <BackCoatings DataType="NameSpan" Preferred="None"/>
        </MediaIntent>
    </ResourcePool>
```

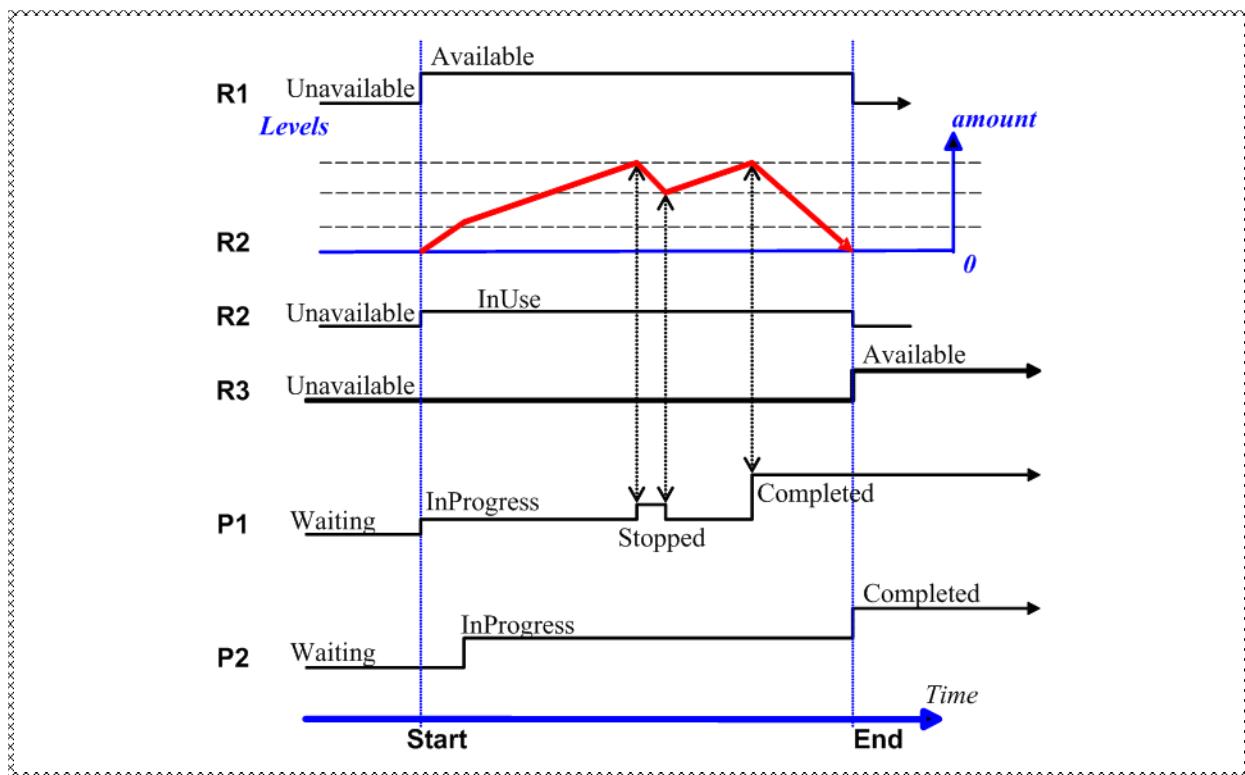
```
<ProductionIntent Class="Intent" ID="ID_PI" Status="Available">
    <ScreeningParamsRef rRef="ScreenID"/>
</ProductionIntent>
<ScreeningParams Class="Parameter" ID="ScreenID" Status="Incomplete">
    <ScreenSelector ScreeningFamily="My favorite screen"
        SpotFunction="Ellipse"/>
</ScreeningParams>
</ResourcePool>
<ResourceLinkPool>
    <ComponentLink Usage="Output" rRef="Link0003"/>
    <LayoutIntentLink Usage="Input" rRef="Link0004"/>
    <MediaIntentLink Usage="Input" rRef="Link0005"/>
    <ProductionIntentLink Usage="Input" rRef="ID_PI"/>
</ResourceLinkPool>
</JDF>
```

JMF

Figure 4-1: Example of a simple Process chain linked by Resources



4.0.1 Partial Processing of Nodes with Partitioned Resources Overlapping Processing Using Pipes

Figure 4-2: Example of status transitions in case of overlapping Processing

```
XJDF<AncestorPool>
  <Ancestor FileName="file:///grandparent.jdf" NodeID="p_01"/>
  <Ancestor FileName="file:///parent.jdf" NodeID="p_02"/>
</AncestorPool>
```

4.0.2

~~XJDF XJMF An XJDF~~

XJMF

Chapter 5 XJMF Messaging with the Job Messaging Format

Introduction A workflow is a dynamic set of interacting Controllers and Devices. For the workflow to run efficiently, these Controllers and Devices need to communicate and interact in a well defined manner. Whereas XJDF will typically be submitted to a Device and only be returned after the process has been executed, XJMF messages MAY be exchanged at any time. Typical use cases for XJMF include but are not limited to:

- System setup
- Dynamic status and error tracking for Jobs and Devices
- Pipe control
- Device setup and Job changes
- Queue handling and Job submission
- Device Capability description

This chapter specifies the XML structure of XJMF. For details of the exchange protocol and data packaging, see section [Section 11.4, “JDF XJDF and JMF XJMF Interchange Protocol” on page 1127](#) and [See Section 11.6, “JDF XJDF Packaging” on page 1134](#)

XJMF Modification Notes

The root element of the message package has been renamed from JMF to XJMF. This allows immediate identification of XJMF and aligns closely with XJDF.

Two JMF Families have been removed:

- 1 Registrations, i.e the request to the recipient of the Registration to send **Command** Messages to a **Command** recipient that is specified in **Subscription**. Registrations have been replaced by **Command** elements with embedded subscriptions. This follows the same model as **Query** elements with embedded subscriptions.
- 2 Acknowledges, i.e. asynchronous responses. The only valid asynchronous response is a **Signal** that may be subscribed to. Note that this does require all **Queue** submissions to be handled synchronously, but this has also been the case in JMF, where the **Command/@AcknowledgeURL** could be omitted, thus forcing the recipient to handle the message synchronously.

Message/@Type has been replaced by an explicit message element that is structured as the combination of message Family and Type. For instance

Example 5-1: Message Type vs Explicit Message Element

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="JDFEditor"TimeStamp="2014-09-18T17:13:18+02:00"
      Version="1.5" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="JMFRootMessage">
  <Query ID="m_000000" Type="KnownMessages" xsi:type="QueryKnownMessages"/>
</JMF>
```

Is now encoded as:

```
<XJMF MaxVersion="2.0" SenderID="JDFEditor"
     TimeStamp="2014-09-18T17:10:29+02:00" Version="2.0" xmlns="http://www.CIP4.org/
      JDFSchema_2_0">
  <QueryKnownMessages ID="m_000000"/>
</XJMF>
```

This naming element structure allows a much cleaner XML schema definition and specification of the respective message elements.

Individual QueueEntry modification messages have been combined into a single ModifyQueueEntry Message.

All Pipe control messages have been combined into a single PipeControl Message.

5.1 XJMF Root

XJMF and XJDF have inherently different structures. In order to allow immediate identification of Messages, XJMF uses the unique name **XJMF** as its own root-element name. XJMF elements SHALL contain one or more messages that provide more detailed information. an **XJDF**

Table 5-1: XJMF Element

Name	Data Type	Description
<i>AgentName</i> ?	string	The name of the application that generated the XJMF . Both the company name and the product name MAY appear, and SHOULD be consistent between versions of the application.
<i>AgentVersion</i> ?	string	The version of the application that generated the XJMF . The format of the version string MAY vary from one application to another, but SHOULD be consistent for an individual application.
<i>DeviceID</i>	NMTOKEN	String that identifies the sender Device. <i>@DeviceID</i> SHALL match <i>Device/@DeviceID</i> of the Device Element that describes the Sender, e.g. when responding to a KnownDevices query. <i>@DeviceID</i> SHOULD be modified to the proxy Controller's <i>@DeviceID</i> when an XJMF is passed through a proxy.
<i>TimeStamp</i>	dateTime	Time stamp that identifies when the Message was created.
<i>Version</i>	JDFJMFVersion	Text that identifies the version of the JMF_Message . The current version of this specification is "2.0".
<i>xmlns</i>	URI	XJDF supports use of XML namespaces. The XJDF namespace SHALL be declared. The value that applies to XJMF that adhere to this specification is "http://www.CIP4.org/XJDFSchema_2_0". For details on using namespaces in XML, see [XMLNS].
<i>Message</i> +	element	Message Element(s). One or more messages SHALL be provided. For a list of Messages, see Section 5.2, "List of All XJMF Messages". The recipient SHALL process the Messages in XML order.

5.1.1 Message

The following table describes the contents of a Message. A Message is an abstract data type. Section 5.2, "List of All XJMF Messages" provides a list of all message element instances.

Table 5-2: Message (Sheet 1 of 2)

Name	Data Type	Description
<i>AgentName</i> ?	string	The name of the application that generated the JMF . Both the company name and the product name MAY appear, and SHOULD be consistent between versions of the application. If not specified, defaults to the value of JMF/@AgentName .

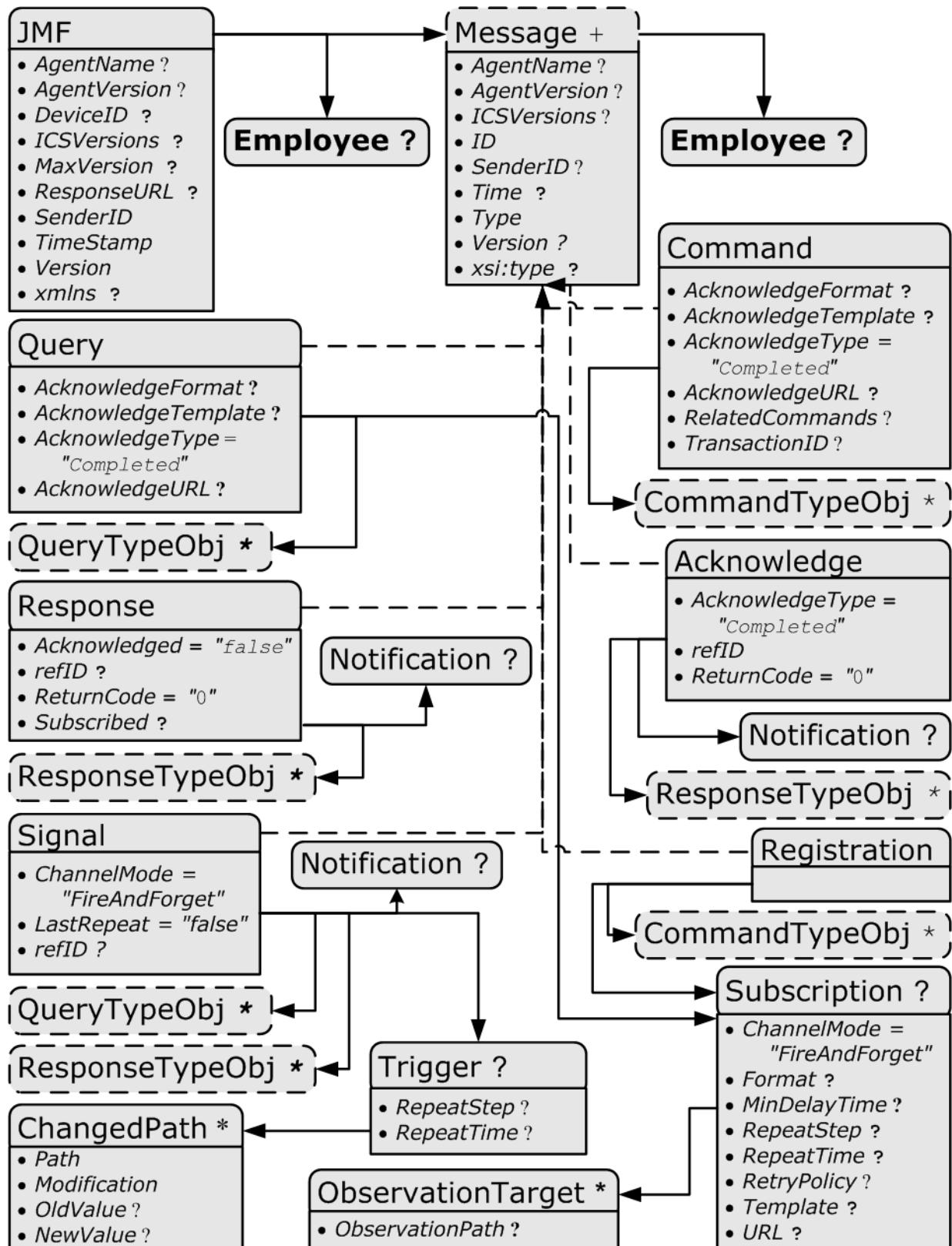
Table 5-2: Message (Sheet 2 of 2)

Name	Data Type	Description
<i>AgentVersion</i> ?	string	The version of the application that generated the JMF. The format of the version string MAY vary from one application to another, but SHOULD be consistent for an individual application. If not specified, defaults to the value of <i>JMF/@AgentVersion</i> .
<i>Author</i> ?	string	Human readable description of the employee that entered the message.
<i>DeviceID</i> ?	NMTOKEN	<i>@DeviceID</i> of the original sender of this Message Element. <i>@DeviceID</i> SHALL NOT be modified when a JMF is passed through a proxy. See also <i>JMF/@DeviceID</i> in Table 5-1.
<i>ICSVersions</i> ?	NMTOKENS	CIP4 Interoperability Conformance Specification (ICS) Versions that this JMF Message complies with. The semantics are identical to <i>JDF/@ICSVersions</i> . If not specified, defaults to the value of <i>JMF/@ICSVersions</i> . Values include those from: <i>JDF/@ICSVersions</i> (Table 3-1, “XJDF Node” on page 50).
<i>PersonalID</i> ?	NMTOKEN	Identifier of the employee that entered the message.
<i>Time</i> ?	dateTime	Time at which the Message was generated. This Attribute NEED NOT be specified unless this time is different from the time specified in the <i>@TimeStamp</i> Attribute of the JMF Element. Note: when a proxy forwards Messages and creates a new JMF parent for a Message, it SHALL update <i>@Time</i> to the value of the original <i>JMF/@TimeStamp</i> if <i>@Time</i> is not provided in the original Message.

5.1.2 Structure Diagram

The following figure depicts the basic XJMF messaging structure and the Message Families. Dashed boxes show abstract objects.

Figure 5-1: JMF Root Element – a diagram of its structure



5.2 List of All XJMF Messages

Table 5-3: List of XJMF Messages (Sheet 1 of 2)

Messages	Description
CommandFlushResources QueryFlushResources ResponseFlushResources SignalFlushResources	Remove temporary Resource from a Device.
CommandForceGang ResponseForceGang	A gang is forced to execute.
CommandGangStatus ResponseGangStatus	The status of a gang is queried.
QueryKnownDevices ResponseKnownDevices SignalKnownDevices	Returns information about the Devices that are controlled by a Controller.
QueryKnownMessages ResponseKnownMessages SignalKnownMessages	Returns a list of all Messages that are supported by the Controller.
QueryKnownSubscriptions ResponseKnownSubscriptions SignalKnownSubscriptions	Returns a list of active persistent channels.
CommandModifyQueueEntry ResponseModifyQueueEntryl	Modifies the properties of one or more QueueEntry elements.
QueryNotification ResponseNotification SignalNotification	Used to signal usual events due to any activities of a Device, operator, etc. (e.g., scanning a bar code).
CommandPipeControl ResponsePipeControl	All pipe related commands are implemented using the PipeControl message.
QueryQueueStatus ResponseQueueStatus SignalQueueStatus	Returns the Queue Elements that describe a queue or set of queues.
CommandRequestQueueEntry ResponseRequestQueueEntryl	A new Job is requested by the Device. This Message is used to signal that a Device has processing Resources available.
CommandResource QueryResource ResponseResource SignalResource	Queries and/or modifies XJDF Resources that are used by a Device, such as Device settings, or by a Job. This Message can also be used to query the level of Resource Elements in a Device.
CommandResubmitQueueEntry ResponseResubmitQueueEntryl	Replaces a queue entry without affecting the entry's parameters. The command is used, for example, for late changes to a submitted XJDF.
CommandReturnQueueEntry ResponseReturnQueueEntryl	Returns a Job that had been submitted with a SubmitQueueEntry to the queue that represents the Controller that originally submitted the Job.
CommandShutDownResponseShutDown	Shuts down a Device.

Table 5-3: List of XJMF Messages (Sheet 2 of 2)

Messages	Description
QueryStatus	Queries the general status of a Device, Controller or Job.
ResponseStatus	
SignalStatus	
CommandStopPersistentChannel	Closes a persistent channel.
ResponseStopPersistentChannel	
CommandSubmitQueueEntry	A Job is submitted to a queue in order to be executed.
ResponseSubmitQueueEntry	
CommandWakeUp	Wakes up a Device that is in standby mode.
ResponseWakeUp	

5.3 XJMF Message Families

A Message belongs to one of four message families. These families are Query, Command, Signal and Response. An explanation of each family is provided in the following sections.

5.3.1 Query

A **Query** is a Message that retrieves information from a Receiver without changing the state of that Receiver. For details of XJMF handshaking, see Section 11.5, “XJMF/XJMF Handshaking”.

XJMF. The Receiver SHALL synchronously respond to a **Query** with an XJMF that contains an Abstract Response of the same type as the **Query**. The following table shows the content of a **Query** Message Element:

Table 5-4: Query

Name	Data Type	Description
AgentName?	string	See <code>Message/@AgentName</code> .
AgentVersion?	string	See <code>Message/@AgentVersion</code> .
Author?	string	See <code>Message/@Author</code> .
DeviceID?	NMTOKEN	See <code>Message/@DeviceID</code> .
ICSVersions?	NMTOKENS	See <code>Message/@ICSVersions</code> .
ID	ID	Identifies the Message.
PersonalID?	NMTOKEN	See <code>Message/@PersonalID</code> .
Time?	dateTime	See <code>Message/@Time</code> .
Subscription?	element	If specified creates a persistent channel. For the structure of a Subscription Element, see Section 5.4.4, “Persistent Channels”.

5.3.1.1 Element: Subscription

A **Subscription** creates a persistent channel. See Section 11.5.2, “Subscribing for Signals” for details.

Table 5-5: Subscription Element

Name	Data Type	Description
<i>ChannelMode</i> ?	enumerations	Specifies reliability of persistent channel, and whether it is required or just preferred. Ordered list, with most preferred channel mode first. If none of the provided values of <i>@ChannelMode</i> are supported by the consumer of the subscription, the Response should indicate <i>@ReturnCode</i> 111, which is “Subscription request denied”. Allowed values are from: Signal/ <i>@ChannelMode</i> . See Table 5.3.3, “Signal”.
<i>RepeatTime</i> ?	double	Requests an update signal every <i>@RepeatTime</i> seconds. If defined, the signal is generated periodically independent of any other trigger conditions.
<i>RetryPolicy</i> ?	enumeration	For reliable subscriptions. Indicates whether or not signals should be retried indefinitely, or only until the next Signal from the same Subscription (i.e., has the same <i>@refID</i>) would be sent. <i>@RetryPolicy</i> is ignored for non-reliable subscriptions. Allowed values are: <i>DiscardAtNextSignal</i> – if a Signal has not been received, and it is time to send the next Signal related to this Subscription (the next Signal specifies the same <i>@refID</i> value), then discard the current Signal. <i>RetryForever</i> – Continue retrying every Signal indefinitely.
<i>URL</i> ?	URL	URL of the persistent channel receiving end. The protocol of the Subscription is specified by the scheme of <i>@URL</i> .

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="Controller-1"
     TimeStamp="2005-07-25T11:38:23+02:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Query ID="M007" Type="KnownDevices" xsi:type="QueryKnownDevices"/>
</JMF>
```

5.3.2 Command

A **Command** is syntactically equivalent to a **Query**, but rather than simply retrieving information, it also causes a state change in the. A **Command** Element is used to change the state of the Receiver. The Receiver SHALL synchronously respond to a **Command** with an XJMF that contains a **Response** of the same type as the **Command**. For details of XJMF handshaking, see Section 11.5, “XJMF/XJMF Handshaking”. The following table contains the contents of a **Command** Message.

Table 5-6: Command Family (Sheet 1 of 2)

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/ <i>@AgentName</i> .
<i>AgentVersion</i> ?	string	See Message/ <i>@AgentVersion</i> .
<i>Author</i> ?	string	See Message/ <i>@Author</i> .
<i>DeviceID</i> ?	XNMTOKEN	See Message/ <i>@DeviceID</i> .
<i>ICSVersions</i> ?	XNMTOKENS	See Message/ <i>@ICSVersions</i> .

Table 5-6: Command Family (Sheet 2 of 2)

Name	Data Type	Description
<i>ID</i>	ID	Identifies the Message.
<i>PersonalID?</i>	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>RelatedCommands?</i>	NMTOKENS	A list of <i>Command/@ID</i> values that need to be processed as a single transaction (in other words all Commands needs to succeed or all need to be rejected). The Commands SHALL be processed in the order specified by this attribute. This attribute SHALL only appear in the last Command of a transaction. An application SHOULD wait for a reasonable amount of time to collect all related Commands prior to failing a transaction.
<i>Time?</i>	dateTime	See <i>Message/@Time</i> .
<i>TransactionID?</i>	NMTOKEN	The ID on the transaction the Command belongs to. All Commands with the same <i>@TransactionID</i> SHALL either all succeed or all fail

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" DeviceID="A3 Printer"
      SenderID="MIS master A" TimeStamp="2000-07-25T12:32:48+02:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Command ID="M009" Type="ResumeQueueEntry" xsi:type="CommandResumeQueueEntry">
    <QueueEntryDef QueueEntryID="job-0032"/>
  </Command>
</JMF>
```

Example 5-2: ResumeQueueEntry Response Message

The following example shows a possible Response Message to the Command Message example above:

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="A3 Printer"
      TimeStamp="2000-07-25T12:32:48+02:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Response ID="M109" Type="ResumeQueueEntry" xsi:type="ResponseResumeQueueEntry"
            refID="M009">
    <Queue DeviceID="A3 Printer" Status="Full">
      <QueueEntry JobID="job-0032" QueueEntryID="job-0032" Status="Running"/>
    </Queue>
  </Response>
</JMF>
```

5.3.3 Signal

A Signal is a Message that a Receiver of a Query with a Subscription SHALL asynchronously send whenever the conditions specified in the Subscription are true.

Note: Signals are typically sent from a Device to a Controller. For details of XJMF handshaking, see Section 11.5, “XJMF/XJMF XJMF/XJMF Handshaking”. For details of setting up subscriptions for Signals, see Section 11.5.2, “Subscribing for Signals”. The following table shows the contents of a Signal.

Table 5-7: Signal Family

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@ <i>AgentName</i> .
<i>AgentVersion</i> ?	string	See Message/@ <i>AgentVersion</i> .
<i>Author</i> ?	string	See Message/@ <i>Author</i> .
<i>ChannelMode</i> ?	enumeration	Specifies reliability of the signal. Allowed values are: <i>FireAndForget</i> – the receiver of the Signal MAY respond using a JMF Response Message. <i>Reliable</i> – Indicates that the Signal is the result of a subscription where reliable signaling was specified in the Subscription Element. The receiver of the Signal SHALL respond using a JMF Response Message.
<i>DeviceID</i> ?	NMTOKEN	See Message/@ <i>DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ <i>ICSVersions</i> .
<i>ID</i>	ID	Identifies the Message.
<i>PersonalID</i> ?	NMTOKEN	See Message/@ <i>PersonalID</i> .
<i>refID</i> ?	NMTOKEN	Identifies the initiating Query Message that subscribed this Signal Message. Hard-wired signals SHALL NOT contain a @ <i>refID</i> Attribute.
<i>Time</i> ?	dateTime	See Message/@ <i>Time</i> .

Example 5-3: Signal Message

The following is an example of a Signal Message:

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="Press 45"
     TimeStamp="2005-07-25T12:28:01+02:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<Signal ID="s123" Type="Status" xsi:type="SignalStatus">
    <StatusQuParams JobID="42" JobPartID="66"/>
    <DeviceInfo DeviceStatus="Setup"/>
</Signal>
</JMF>
```

5.3.4 Response

A Response is a Message that a Receiver SHALL synchronously send to a Sender as a response to a Message. For details of XJMF handshaking, see Section 11.5, “XJMF/XJMF Handshaking”.

The following table shows the content of a Response Message.

Table 5-8: Response Family

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@ <i>AgentName</i> .
<i>AgentVersion</i> ?	string	See Message/@ <i>AgentVersion</i> .
<i>Author</i> ?	string	See Message/@ <i>Author</i> .
<i>DeviceID</i> ?	NMTOKEN	See Message/@ <i>DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ <i>ICSVersions</i> .
<i>ID</i> ?	ID	Identifies the Message.
<i>PersonalID</i> ?	NMTOKEN	See Message/@ <i>PersonalID</i> .
<i>refID</i> ?	NMTOKEN	Copy of the @ <i>ID</i> Attribute of the initiating Message to which the Response refers. If not specified, the Response Message refers to the entire XJMF(e.g., if the XJMF was not parseable).
<i>ReturnCode</i> ?	integer	@ <i>ReturnCode</i> summarizes the result of the Response. The value "0" indicates success. @ <i>ReturnCode</i> SHALL be provided if an error occurred. For predefined values see Appendix D, "Supported Error Codes in XJMF and Notification Elements" on page 1183. Note: additional values MAY be specified in ICS documents.
<i>Subscribed</i> ?	boolean	If a Subscription Element has been supplied by the corresponding query, this Attribute indicates whether the Subscription has been refused or accepted. If "true", the requested Subscription is accepted. If "false", the Subscription is refused because the Controller does not support persistent channels. For details, see Section 5.4.4, "Persistent Channels".
<i>Time</i> ?	dateTime	See Message/@ <i>Time</i> .
<i>Notification</i> ?	element	Additional information including textual description of the return code. The Notification Element SHOULD be provided if the @ <i>ReturnCode</i> is greater than 0, which indicates that an error has occurred. See Section 5.5.6.3.1, "Notification".

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="RIP-1"
     TimeStamp="2000-07-25T11:38:25+02:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Response ID="M107" Type="KnownDevices" xsi:type="ResponseKnownDevices"
            refID="M007">
    <DeviceList>
      <DeviceInfo DeviceStatus="Unknown">
        <Device DeviceID="Rip1"/>
      </DeviceInfo>
      <DeviceInfo DeviceStatus="Unknown">
        <Device DeviceID="Rip2"/>
      </DeviceInfo>
    </DeviceList>
  </Response>
</JMF>

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="A3 Printer"
     TimeStamp="2000-07-25T12:32:48+02:00" MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

```

```

<Acknowledge ID="M109" Type="PipePush" xsi:type="AcknowledgePipePush" refID="M010">
    <JobPhase JobID="J1" JobPartID="1" Status="InProgress"/>
</Acknowledge>
</JMF>

```

an XJDF

```

or a SHOULD SHOULD <JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="A3 Printer"
    TimeStamp="2013-03-25T12:32:48+02:00"
    MaxVersion="1.5" Version="1.5"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<Response ID="M109" ReturnCode="5" Type="ResumeQueueEntry"
    xsi:type="ResponseResumeQueueEntry" refID="M009">
    <Notification Class="Error" TimeStamp="2005-03-25T12:32:48+02:00" Type="Error">
        <Comment>StartJob unsuccessful - Device does not handle commands</Comment>
        <Error ErrorID="1234" Resend="Prohibited">
            <ErrorData Path="/JMF/Command" ErrorType="Unsupported"/>
        </Error>
    </Notification>
</Response>
</JMF>

```

5.4 Message Template

The previous sections in this chapter provide a description of the overall structure of JMF Messages. This section contains a list of the standard Messages that are defined within the **XJDF** framework. It is OPTIONAL for an **XJDF**-compliant application to support each Message described in this list.

At the beginning of each section there is a table that lists all of the Message types in that category. These tables contain three columns. The first is entitled “Message Type,” and it lists the names of each Message type. The second column is entitled “Family.” The values in this (family) column describe the kind of Message Element that is applicable in the circumstance being illustrated. The following abbreviations are used to describe the values used in the tables below to describe these major Message Element types.

C:	Command
Q:	Query
R:	Response
S:	Signal

Each explicit message element will be described in the respective message section. In case of traits that are inherited from abstract Command, Query, Signal or Response elements, the descriptions will be referenced.

5.5 Messages for Events and Capabilities

The Message types of the following table are defined in order to exchange metadata about Controller or Device abilities and for general communication.

Table 5-9: Messages for events and capabilities (Sheet 1 of 2)

Messages	Description
QueryKnownDevices	Returns information about the Devices that are controlled by a Controller.
ResponseKnownDevices	
SignalKnownDevices	
QueryKnownMessages	Returns a list of all Messages that are supported by the Controller.
ResponseKnownMessages	
SignalKnownMessages	

Table 5-9: Messages for events and capabilities (Sheet 2 of 2)

Messages	Description
QueryKnownSubscriptions	Returns a list of active persistent channels.
ResponseKnownSubscriptions	
SignalKnownSubscriptions	
QueryNotification	Generally sent as Signals. A Query allows Subscriptions for Notification Messages.
ResponseNotification	
SignalNotification	
CommandStopPersistentChannel	Closes a persistent channel.
ResponseStopPersistentChannel	

5.5.1

5.5.2

5.5.3 KnownDevices

The KnownDevices Query Message requests information about the Devices that are controlled by a Controller. If a high level Controller controls lower level Controllers, it SHOULD also list the Devices that are controlled by these. The response is a DeviceList which is list of DeviceInfo Elements controlled by the Controller that receives the query, as demonstrated in Example 5-4.

Example 5-4: KnownDevices Response

TBD 2.x Example.

```
<Response ID="M1" refID="Q1" Type="KnownDevices" xsi:type="ResponseKnownDevices">
  <DeviceList>
    <DeviceInfo DeviceStatus="Unknown">
      <Device DeviceID="Joe SpeedMaster"
             DeviceType="Heidelberg SM102/6 rev. 47"/>
    </DeviceInfo>
  </DeviceList>
</Response>
```

5.5.3.1 QueryKnownDevices

Table 5-10: QueryKnownDevices Message (Sheet 1 of 2)

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Query/@ID.

Table 5-10: QueryKnownDevices Message (Sheet 2 of 2)

Name	Data Type	Description
PersonalID?	NMTOKEN	See Message/@PersonalID.
Time?	dateTime	See Message/@Time.
DeviceFilter?	element	Refines the list of Devices queried. Only Devices that match the DeviceFilter are listed. The default is to return a list of all known Devices.
Subscription?	element	See Query/Subscription.

5.5.3.1.1 Element: DeviceFilter

The DeviceFilter Element refines the list of Devices that are requested to be returned. Only Devices that match all parameters of one of the **Device** Resources specified in the DeviceFilter Element are included.

Table 5-11: DeviceFilter Element

Name	Data Type	Description
DeviceDetails?	enumeration	<p>Refines the level of provided information about the Device.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>None</i> – Provide only DeviceInfo/@DeviceID and DeviceInfo/@DeviceStatus. <i>Brief</i> – Provide all available Device information except for Device Elements. <i>Modules</i> – ModuleStatus Elements are to be provided without module specific status details and without module specific employee information. <i>Details</i> – Provide maximum available Device information excluding Device capability descriptions. Includes Device Elements which represent details of the Device. <i>Full</i> – Provide maximum available Device information including Device capability descriptions. Includes Device Elements which represent details of the Device.

5.5.3.2 ResponseKnownDevices

Table 5-12: ResponseKnownDevices Message (Sheet 1 of 2)

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID?	ID	See Response/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
refID?	NMTOKEN	See Response/@refID.
ReturnCode?	integer	See Response/@ReturnCode.
Subscribed?	boolean	See Response/@Subscribed.

Table 5-12: ResponseKnownDevices Message (Sheet 2 of 2)

Name	Data Type	Description
Time ?	dateTime	See Message/@Time.
DeviceList ?	element	The list of known Devices.
Notification ?	element	See Response/Notification.

5.5.3.2.1 Element: DeviceList

The DeviceList Element contains a list of information about Devices that are returned.

Table 5-13: DeviceList Element

Name	Data Type	Description
DeviceInfo *	element	List of information about known Devices as requested by the DeviceFilter Element. For details of the DeviceInfo Element, see Table 5-57, “DeviceInfo Element” on page 247 in the Message description Section 5.6.4, “Status”.

5.5.3.3 SignalKnownDevices

Table 5-14: SignalKnownDevices Message

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
ChannelMode ?	enumeration	See Signal/@ChannelMode.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Signal/@ID.
PersonalID ?	NMTOKEN	See Message/@PersonalID.
refID ?	NMTOKEN	See Signal/@refID.
Subscribed ?	boolean	See Query/@Subscribed.
Time ?	dateTime	See Message/@Time.
DeviceList ?	element	The list of known Devices.

5.5.4 KnownMessages

The KnownMessages Query Message returns a list of all Message types that are supported by the Controller.

5.5.4.1 QueryKnownMessages

Table 5-15: QueryKnownMessages Message (Sheet 1 of 2)

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.

Table 5-15: QueryKnownMessages Message (Sheet 2 of 2)

Name	Data Type	Description
<i>DeviceID</i> ?	NMTOKEN	See <i>Message/@DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See <i>Message/@ICSVersions</i> .
<i>ID</i>	ID	See <i>Query/@ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>Time</i> ?	dateTime	See <i>Message/@Time</i> .
<i>KnownMsgQuParams</i> ?	element	Refines the query for known Messages. If not specified, list all supported Message types.
<i>Subscription</i> ?	element	See <i>Query/Subscription</i> .

5.5.4.1.1 Element: KnownMsgQuParams

KnownMsgQuParams is a filter that specifies the details of the returned *MessageService* elements.

Table 5-16: KnownMsgQuParams Element

Name	Data Type	Description
<i>Exact</i> ?	boolean	Requests an exact description of the known Messages. If "true", the response also contains a <i>FileSpec</i> (Schema) that references a schema.

5.5.4.2 ResponseKnownMessages

Table 5-17: ResponseKnownMessages Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See <i>Message/@AgentName</i> .
<i>AgentVersion</i> ?	string	See <i>Message/@AgentVersion</i> .
<i>Author</i> ?	string	See <i>Message/@Author</i> .
<i>DeviceID</i> ?	NMTOKEN	See <i>Message/@DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See <i>Message/@ICSVersions</i> .
<i>ID</i> ?	ID	See <i>MessageResponse/@ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>refID</i> ?	NMTOKEN	See <i>Response/@refID</i> .
<i>ReturnCode</i> ?	integer	See <i>Response/@ReturnCode</i> .
<i>Subscribed</i> ?	boolean	See <i>Response/@Subscribed</i> .
<i>Time</i> ?	dateTime	See <i>Message/@Time</i> .
<i>MessageService</i> *	element	Specifies the supported Messages. Multiple <i>MessageService</i> Elements MAY be specified for a Message with a given JMF/@Type.
<i>Notification</i> ?	element	See <i>Response/Notification</i> .

5.5.4.2.1 Element: MessageService

The response is a list of *MessageService* Elements, one for each supported Message type. The flags of the *MessageService* Response Message Element are set in each *MessageService* entry. They define the supported usage of the Message by the Controller. Note that no @Response Attribute is included in the list, since the capability to

process one of the other Message Families implies the capability to generate an appropriate Response Message. Multiple flags are allowed.

Table 5-18: MessageService Element

Name	Data Type	Description
<i>ChannelMode</i> ?	enumerations	Specifies the supported channel modes for the Message. Allowed values are from: Signal/@ <i>ChannelMode</i> . See Table 5-7, “Signal Family”.
<i>JMFRole</i> ?	enumeration	The role of the Device that responds with the MessageService. Allowed values are: <i>Receiver</i> – The Device that responds to KnownMessages receives and responds to the Message specified in @ <i>Type</i> . This MessageService specifies Query Messages, Signal Messages, Command Messages that the Device understands. <i>Sender</i> – The Device that responds to KnownMessages is the originator of the Message specified in @ <i>Type</i> . This MessageService specifies Response Elements.
<i>Type</i>	NMTOKEN	Name of the supported Message element, e.g. QueryKnownMessages. Additional information MAY be specified in FileSpec(Schema).
<i>URLSchemes</i> ?	NMTOKENS	List of schemes supported for the Message defined by this MessageService. Values include: <i>file</i> – The file scheme according to [RFC1738] and [RFC3986]. <i>http</i> – HTTP (Hypertext Transport Protocol) <i>https</i> – HTTPS (Hypertext Transport Protocol – Secure)
<i>FileSpec</i> (Schema) ?	element	Reference to an XML schema description of the XJMF capabilities of the Device.

Example 5-5: KnownMessages Response

The following is an example of a Response Message to a KnownMessages Query Message:

```
<Response ID="M1" Type="KnownMessages" xsi:type="ResponseKnownMessages" refID="Q1">
  <MessageService JMFRole="Receiver" Query="true" Type="KnownMessages"/>
  <MessageService JMFRole="Receiver" Persistent="true" Query="true" Signal="true"
    Type="Status"/>
</Response>
```

5.5.4.3 SignalKnownMessages

Table 5-19: SignalKnownMessages Message (Sheet 1 of 2)

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@ <i>AgentName</i> .
<i>AgentVersion</i> ?	string	See Message/@ <i>AgentVersion</i> .
<i>Author</i> ?	string	See Message/@ <i>Author</i> .
<i>ChannelMode</i> ?	enumeration	See Signal/@ <i>ChannelMode</i> .
<i>DeviceID</i> ?	NMTOKEN	See Message/@ <i>DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ <i>ICSVersions</i> .

Table 5-19: SignalKnownMessages Message (Sheet 2 of 2)

Name	Data Type	Description
<i>ID</i>	ID	See <i>Signal/@ID</i> .
<i>PersonalID?</i>	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>refID?</i>	NMTOKEN	See <i>Signal/@refID</i> .
<i>Subscribed?</i>	boolean	See <i>Query/@Subscribed</i> .
<i>Time?</i>	dateTime	See <i>Message/@Time</i> .
<i>MessageService</i> *	element	Specifies the supported Messages. Multiple <i>MessageService</i> Elements MAY be specified for a Message with a given JMF/@Type.

5.5.5 KnownSubscriptions

The KnownSubscriptions XJMF enables Controllers to query Devices for a list of active persistent channels. Job-specific subscriptions are not supported.

5.5.5.1 QueryKnownSubscriptions

Table 5-20: QueryKnownSubscriptions Message

Name	Data Type	Description
<i>AgentName?</i>	string	See <i>Message/@AgentName</i> .
<i>AgentVersion?</i>	string	See <i>Message/@AgentVersion</i> .
<i>Author?</i>	string	See <i>Message/@Author</i> .
<i>DeviceID?</i>	NMTOKEN	See <i>Message/@DeviceID</i> .
<i>ICSVersions?</i>	NMTOKENS	See <i>Message/@ICSVersions</i> .
<i>ID</i>	ID	See <i>Query/@ID</i> .
<i>PersonalID?</i>	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>Time?</i>	dateTime	See <i>Message/@Time</i> .
<i>Subscription?</i>	element	See <i>Query/Subscription</i> .
<i>SubscriptionFilter?</i>	element	Refines the query for known Messages. If not specified, list all supported Message types.

5.5.5.1.1 Element: SubscriptionFilter

The SubscriptionFilter Element is a filter to limit the list of SubscriptionInfo Elements that are returned in the KnownSubscriptions Response.

Table 5-21: SubscriptionFilter Element (Sheet 1 of 2)

Name	Data Type	Description
<i>ChannelID?</i>	NMTOKEN	@ChannelID of the persistent channel to be queried. If the channel has been created with a Query Message, the @ChannelID specifies the ID of the Query Message (identical to the @refID of the Response Message)
<i>DeviceID?</i>	NMTOKEN	Only subscription from Devices or Controllers with a matching @DeviceID Attribute are queried

Table 5-21: SubscriptionFilter Element (Sheet 2 of 2)

Name	Data Type	Description
<i>MessageTypes</i> ?	NMTOKENS	List of supported Message element names of the subscribed messages. If not specified, <i>SubscriptionInfo</i> elements are returned for all message types.
<i>URL</i> ?	URL	URL of the receiving Controller. This SHALL be identical to the URL that was used to create the persistent channel. If no <i>@ChannelID</i> is specified, all persistent channels to this <i>@URL</i> are queried.

5.5.5.2 ResponseKnownSubscriptions

Table 5-22: ResponseKnownSubscriptions Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See <i>Message/@AgentName</i> .
<i>AgentVersion</i> ?	string	See <i>Message/@AgentVersion</i> .
<i>Author</i> ?	string	See <i>Message/@Author</i> .
<i>DeviceID</i> ?	NMTOKEN	See <i>Message/@DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See <i>Message/@ICSVersions</i> .
<i>ID</i> ?	ID	See <i>Response/@ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>refID</i> ?	NMTOKEN	See <i>Response/@refID</i> .
<i>ReturnCode</i> ?	integer	See <i>Response/@ReturnCode</i> .
<i>Subscribed</i> ?	boolean	See <i>Response/@Subscribed</i> .
<i>Time</i> ?	dateTime	See <i>Message/@Time</i> .
<i>Notification</i> ?	element	See <i>Response/Notification</i> .
<i>SubscriptionInfo</i> *	element	Refines the query for known Messages. If not specified, list all supported Message types.

5.5.5.2.1 Element: SubscriptionInfo

A *SubscriptionInfo* Element describes the details of existing subscriptions.

Table 5-23: SubscriptionInfo Element

Name	Data Type	Description
<i>ChannelID</i>	NMTOKEN	<i>@ChannelID</i> specifies the ID of the Query message that initiated the Subscription. <i>@ChannelID</i> SHALL match <i>@refID</i> of all Signals that comprise this persistent channel.
<i>DeviceID</i>	NMTOKEN	<i>@DeviceID</i> of the controller that subscribed for the persistent channel. <i>@DeviceID</i> SHALL match <i>@DeviceID</i> of the query that subscribed for this persistent channel.
<i>MessageType</i>	NMTOKEN	<i>@MessageType</i> SHALL match the local element name (i.e. without namespace prefix) of the Signals that comprise this persistent channel.
<i>Subscription</i>	element	The <i>Subscription</i> Element that describes the persistent channel.

5.5.5.3 SignalKnownSubscriptions

Table 5-24: SignalKnownSubscriptions Message

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName .
AgentVersion ?	string	See Message/@AgentVersion .
Author ?	string	See Message/@Author .
ChannelMode ?	enumeration	See Signal/@ChannelMode .
DeviceID ?	NMTOKEN	See Message/@DeviceID .
ICSVersions ?	NMTOKENS	See Message/@ICSVersions .
ID	ID	See Signal/@ID .
PersonalID ?	NMTOKEN	See Message/@PersonalID .
refID ?	NMTOKEN	See Signal/@refID .
Time ?	dateTime	See Message/@Time .
SubscriptionInfo *	element	Refines the query for known Messages. If not specified, list all supported Message types.

5.5.6 Notification

Notification Messages are generally sent as Signals. The Query is defined to allow subscriptions for Notification Messages.

Notification Elements are also used to signal usual events due to any activities of a Device, operator, etc. (e.g., scanning a bar code).

5.5.6.1 QueryNotification

Table 5-25: QueryNotification Message

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName .
AgentVersion ?	string	See Message/@AgentVersion .
Author ?	String	See Message/@Author .
DeviceID ?	NMTOKEN	See Message/@DeviceID .
ICSVersions ?	NMTOKENS	See Message/@ICSVersions .
ID	ID	See MessageQuery/@ID .
PersonalID ?	NMTOKEN	See Message/@PersonalID .
Time ?	dateTime	See Message/@Time .
NotificationFilter ?	element	Defines the types of Notification Elements that should be returned
Subscription ?	element	See Query/Subscription .

5.5.6.1.1 Element: NotificationFilter

Table 5-26: NotificationFilter Element

Name	Data Type	Description
<i>Classes</i> ?	enumerations	<p>Defines the set of <i>Notification/@Class</i> to be queried/subscribed for.</p> <p>Default behavior: all <i>Notification</i> Classes are subscribed to.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Event</i> <i>Information</i> <i>Warning</i> <i>Error</i> <i>Fatal</i> <p>Constraint note: If the values both <i>@Classes</i> and <i>@Types</i> are lists of values, the <i>NotificationFilter</i> defines an OR of all combinations.</p>
<i>MilestoneTypes</i>	NMTOKENS	<p>Matching Milestone types are returned and/or subscribed to.</p> <p>Default value is: all supported <i>@MilestoneType</i> values.</p> <p>Values include those from: Table C-10, “MessageEvents and MilestoneType Values” on page 1174.</p>
<i>Types</i> ?	NMTOKENS	<p>Matching notification types are returned/subscribed.</p> <p>Default value is: all supported notification types.</p> <p>Values include those from: Table C-11, “List of NotificationDetails Elements” on page 1175.</p>

Example 5-6: Notification Signal

```
<Signal ID="S1" Type="Notification" xsi:type="SignalNotification">
  <Notification Class="Event"TimeStamp="2005-07-25T12:32:48+02:00"
    Type="Barcode">
    <Comment>Palette completed</Comment>
    <Barcode Code="99923AAA123"/>
  </Notification>
</Signal>
```

5.5.6.2 ResponseNotification

Table 5-27: ResponseNotification Message (Sheet 1 of 2)

Name	Data Type	Description
<i>AgentName</i> ?	string	See <i>Message/@AgentName</i> .
<i>AgentVersion</i> ?	string	See <i>Message/@AgentVersion</i> .
<i>Author</i> ?	string	See <i>Message/@Author</i> .
<i>DeviceID</i> ?	NMTOKEN	See <i>Message/@DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See <i>Message/@ICSVersions</i> .
<i>ID</i> ?	ID	See <i>Response/@ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>refID</i> ?	NMTOKEN	See <i>Response/@refID</i> .

Table 5-27: ResponseNotification Message (Sheet 2 of 2)

Name	Data Type	Description
ReturnCode?	integer	See Response/@ReturnCode.
Subscribed?	boolean	See Response/@Subscribed.
Time?	dateTime	See Message/@Time.
Notification?	element	See Response/Notification.

5.5.6.3 SignalNotification

Table 5-28: SignalNotification Message

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
ChannelMode?	enumeration	See Signal/@ChannelMode.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Signal/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
refID?	NMTOKEN	See Signal/@refID.
Time?	dateTime	See Message/@Time.
Notification?	element	Notification that describes the event. See Section 5.5.6.3.1, “Notification” on page 227.

5.5.6.3.1 Element: Notification

This Element contains information about individual events that occurred during processing. For a detailed discussion of event properties, see Section 4.7, “Error Handling”.

Table 5-29: Notification Element (Sheet 1 of 3)

Name	Data Type	Description
AgentName?	string	The name of the application that added the Notification Element to the AuditPool (and was responsible for the creation or modification). Both the company name and the product name MAY appear, and SHOULD be consistent between versions of the application..
AgentVersion?	string	The version of the application that added the Notification Element to the AuditPool (and was responsible for the creation or modification). The format of the version string MAY vary from one application to another, but SHOULD be consistent for an individual application.

Table 5-29: Notification Element (Sheet 2 of 3)

Name	Data Type	Description
<i>Class</i>	enumeration	<p>Class of the notification.</p> <p>Allowed values are (in order of severity from lowest to highest):</p> <ul style="list-style-type: none"> <i>Event</i> – Indicates that a pure event due to certain operation-related activity has occurred (e.g., Machine events, operator activities, etc.). This Class is used for the transfer of conventional event Messages. In case of <i>@Class</i> = "Event", further event information is to be provided by the <i>@Type</i> Attribute and Error, Event and Milestone Elements. <i>Information</i> – Any information about a Process which cannot be expressed by the other Classes (e.g., the beginning of execution). No user interaction is needed. <i>Warning</i> – Indicates that a minor error has occurred, and an automatic fix was applied. Execution continues. This appears in situations such as A4-Letter substitutions when toner is low. <i>Error</i> – Indicates that an error has occurred that requires user interaction. This value appears in situations such as when Resources are missing, when major incompatibilities are detected, or when the toner is empty. <i>Fatal</i> – Indicates that a fatal error led to abortion of the Process. This value is seen with most protocol errors or when major Device malfunction has occurred.
<i>ContactRefs?</i>	IDREFS	The employees associated with this event or creator of it.
<i>ID?</i>	ID	<i>@ID</i> of the Notification. <i>@ID</i> SHALL be specified if there is support to subsequently create correction Notification Elements.
<i>JobID?</i>	NMTOKEN	<i>@JobID</i> that this Notification applies to.
<i>JobPartID?</i>	NMTOKEN	<i>@JobPartID</i> that this Notification applies to.
<i>ModuleID?</i>	NMTOKEN	<i>@ModuleID</i> of the Module that this Notification relates to.
<i>ModuleType?</i>	NMTOKEN	<p>Module description.</p> <p>Values include those from: Section C.2, "ModuleType Supported Strings" on page 1170.</p> <p>Note: the allowed values depend on the type of Device. Each type of Device has a separate table of values.</p>
<i>QueueEntryID?</i>	NMTOKEN	<i>@QueueEntryID</i> of the QueueEntry during which this Notification was generated.
<i>refID?</i>	IDREF	Reference to a previous Notification that this Notification corrects. The referenced Notification SHALL reside in the same AuditPool.
<i>TimeStamp</i>	dateTime	Records the date and time when the Element referred to by this Notification Audit was created.

Table 5-29: Notification Element (Sheet 3 of 3)

Name	Data Type	Description
Type ?	enumeration	<p>Identifies the type of notification.</p> <p>Note: @Type allows parsers that do not have access to the schema to find the instance.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> Error Event Milestone
Comment *	element	A Comment Element contains a verbose, human-readable description of the event. If the value of the @Class Attribute is one of "Information", "Warning", "Error" or "Fatal", at least one Comment Element SHOULD be specified. Otherwise (including for @Class = "Event"), Comment Elements are OPTIONAL.
Error ?	element	See Error element below.
Event ?	element	See Event element below.
Milestone ?	element	See Milestone element below.
Part *	element	Describes which parts of a Process this Notification belongs to. If Part is not specified for a Notification, it refers to all parts. For example, imagine a print Job that is to produce three different Sheets. All Sheets are described by one Partitioned Resource. The Part Elements define, unambiguously, the Sheet to which the Notification refers.

5.5.6.3.2 Element: Error

This Element provides additional information for common errors.

Table 5-30: Error Element

Name	Data Type	Description
ErrorID ?	string	Internal Error ID of the application that declares the error.

5.5.6.3.3 Element: Event

This Element provides additional information for common events.

Table 5-31: Event Element

Name	Data Type	Description
EventID	string	Internal Event ID of the application that emits the event.
EventValue ?	string	Additional user defined value related to this event.

5.5.6.3.4 Element: Milestone

In addition to the concrete XJMF feedback both from production to MIS and MIS to production with respect to finished Processes (see Section 5.6.4, “Status” on page 244) and available/consumed Resources (see Section 5.6.2, “Resource” on page 236), many Actors in the workflow want to track certain overall milestones concerning the entire Job across all Resources and Processes in order to display this to the operator. Sometimes the XJMF recipients cannot determine these milestones from the detailed XJDF/XJMF. Therefore a more abstract representation of Job status is described by Milestone events. Note that Milestone Elements usually refer to events involving multiple objects, although the Milestone/@MilestoneType is specified as a singular. The scope of the Milestone is defined by the parent Notification element

Table 5-32: Milestone Element

Name	Data Type	Description
<i>MilestoneType</i>	NMTOKEN	Type of Milestone. Values include those from: Table C-10, “MessageEvents and MilestoneType Values” on page 1174.
<i>TypeAmount?</i>	integer	Indication of how many Elements have been processed (if the milestone refers to certain Resources) (e.g., number of pages proofed, number of different printed Sheets (not the cumulative amount)).

5.5.7

```

<Command ID="M001" Type="RequestForAuthentication"
  xsi:type="CommandRequestForAuthentication">
  <AuthenticationCmdParams AuthenticationType="AsClient"
    Reason="InitiateConnection">
    <Certificate>=====BEGIN CERTIFICATE=====
MIIC3jCCApwCBEIWY6YwCwYHKoZIzjgEAwUAMFUxCzAJBgNVBAYTAKNIMQ8wDQYDVQQHEwZadXJp
Y2gxDTALBgNVBAoTBENJUDQxDzANBgNVBAsTBkpNRiBXRzEVMBMGA1UEAxMMd3d3LmNpcDQub3Jn
MB4XDTA1MDIxODIxNTIzOFoXDTA1MDUyOTIxNTIzOFowVTELMAkGA1UEBhMCQ0gxDzANBgNVBAct
Blp1cm1jaDENMASGA1UEChMEQ01QNDEPMA0GA1UECxMGSk1GIFdHMRUwEwYDVQQDEwx3d3cuY2lw
NC5vcmcwggG3MIIBLAYHKoZIzjgEATCCAR8CgYEAX9TgR11Ei1S30qcLuzk5/YRt1I870QAwx4/
gLRJmlFXUAIuftZPY1Y+r/F9bow9subVWzXgTuAHTRv8mZgt2uZUKWkn5/oBhsQIsJPu6nX/rfG
G/g7V+fGqKYVDwT7g/bTxR7DAjVUE1oWkTL2dfOuK2HXKu/yIgMZndFIAccCFQCXYFCPFMSMLzLKS
uYKi64QL8Fgc9QKBgQD34aCF1ps93su8q1w2uFe5eZSvu/o66oL5V0wLPQeCZ1FZV4661F1P5nEH
EIGAtEkWcSPoTCgWE7fPCTKMyKbhPBZ6i1R8jsjgo64eK70mdZFuo38L+iE1YvH7YnoBJDvMpPG+
qFGQiaiD3+Fa5Z8GkomXoB7VSvkAUw7/s9JKgOBhAACgYArHi/BVNF3OG0JIIdzWraVrx1wg9RM
do+tYRjY4bxue7LRDCvVaSX1Ddy9kTyeTTntwUrJ0yx/8qEi/WmraGXhK8wGSrtE/q3S/A16DwEB
CiyeMhlCrd4QiAhp5WtR4KIMIBjq2Xn8+0MnnT1qDnmesNaSwdZ/01E0azSPTy5XnDALBgcqhkjo
OAQDBQADLwAwLAIUFZHoJvsO3+UYMBZk6yDzhdejzMCFHC0WbkDwfImQCa+dTebXZ1e1G1Q
=====END CERTIFICATE=====</Certificate>
  </AuthenticationCmdParams>
</Command>

<Response ID="M101" Type="RequestForAuthentication" refID="M001"
  xsi:type="ResponseRequestForAuthentication" ReturnCode="304">
  <AuthenticationResp SecureURL="https://printserver.mycompany.com/A3Printer">
    <Certificate>=====BEGIN CERTIFICATE=====
uYKi64QL8Fgc9QKBgQD34aCF1ps93su8q1w2uFe5eZSvu/o66oL5V0wLPQeCZ1FZV4661F1P5nEH
EIGAtEkWcSPoTCgWE7fPCTKMyKbhPBZ6i1R8jsjgo64eK70mdZFuo38L+iE1YvH7YnoBJDvMpPG+
qFGQiaiD3+Fa5Z8GkomXoB7VSvkAUw7/s9JKgOBhAACgYArHi/BVNF3OG0JIIdzWraVrx1wg9RM
do+tYRjY4bxue7LRDCvVaSX1Ddy9kTyeTTntwUrJ0yx/8qEi/WmraGXhK8wGSrtE/q3S/A16DwEB
CiyeMhlCrd4QiAhp5WtR4KIMIBjq2Xn8+0MnnT1qDnmesNaSwdZ/01E0azSPTy5XnDALBgcqhkjo
OAQDBQADLwAwLAIUFZHoJvsO3+UYMBZk6yDzhdejzMCFHC0WbkDwfImQCa+dTebXZ1e1G1Q
MIIC3jCCApwCBEIWY6YwCwYHKoZIzjgEAwUAMFUxCzAJBgNVBAYTAKNIMQ8wDQYDVQQHEwZadXJp
Y2gxDTALBgNVBAoTBENJUDQxDzANBgNVBAsTBkpNRiBXRzEVMBMGA1UEAxMMd3d3LmNpcDQub3Jn
MB4XDTA1MDIxODIxNTIzOFoXDTA1MDUyOTIxNTIzOFowVTELMAkGA1UEBhMCQ0gxDzANBgNVBAct
Blp1cm1jaDENMASGA1UEChMEQ01QNDEPMA0GA1UECxMGSk1GIFdHMRUwEwYDVQQDEwx3d3cuY2lw
NC5vcmcwggG3MIIBLAYHKoZIzjgEATCCAR8CgYEAX9TgR11Ei1S30qcLuzk5/YRt1I870QAwx4/
gLRJmlFXUAIuftZPY1Y+r/F9bow9subVWzXgTuAHTRv8mZgt2uZUKWkn5/oBhsQIsJPu6nX/rfG
G/g7V+fGqKYVDwT7g/bTxR7DAjVUE1oWkTL2dfOuK2HXKu/yIgMZndFIAccCFQCXYFCPFMSMLzLKS

```

```

=====END CERTIFICATE=====
```

```

</AuthenticationResp>
</Response>

<Query ID="M004" Type="RequestForAuthentication"
       xsi:type="QueryRequestForAuthentication">
  <AuthenticationQuParams AuthenticationType="AsClient"/>
</Query>

<Response ID="M102" Type="RequestForAuthentication" refID="M004"
          xsi:type="ResponseRequestForAuthentication" ReturnCode="0">
</Response>

```

5.5.8 StopPersistentChannel

The StopPersistentChannel Command Message unregisters a listening Controller from a persistent channel. No more Signal Messages are sent to the Controller once the command has been issued. A certain subset of signals MAY be addressed for unsubscription by specifying a StopPersChParams Element.

5.5.8.1 CommandStopPersistentChannel

Table 5-33: CommandStopPersistentChannel Message

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Command/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
RelatedCommands?	NMTOKENS	See Command/@RelatedCommands.
Time?	dateTime	See Message/@Time.
TransactionID?	NMTOKEN	See Command/@TransactionID.
StopPersChParams	element	Specifies the persistent channel and the Message types to be unsubscribed.

5.5.8.1.1 Element: StopPersChParams

If the OPTIONAL Attributes are not specified, those Attributes default to match anything. Therefore, it is possible to cancel the persistent channel for Messages belonging to a certain type of Message or to a certain Job.

Table 5-34: StopPersChParams Element (Sheet 1 of 2)

Name	Data Type	Description
ChannelID?	NMTOKEN	@ChannelID of the persistent channel to be deleted. If the channel has been created with a Query Message, the @ChannelID specifies the @ID of the Query Message (identical to the @refID of the Response Message).

Table 5-34: StopPersChParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>MessageType</i> ?	NMTOKEN	Only Messages with a matching Message type are suppressed. Default value is: all Message types Values include those from: <i>Message/@Type</i> .
<i>DeviceID</i> ?	NMTOKEN	Only Messages from Devices or Controllers with a matching <i>@DeviceID</i> Attribute are suppressed.
<i>URL</i>	URL	URL of the receiving Controller. This SHALL be identical to the URL that was used to create the persistent channel. If no <i>@ChannelID</i> is specified, all persistent channels to this URL are deleted.

5.5.8.2 ResponseStopPersistentChannel

Table 5-35: ResponseStopPersistentChannel Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See <i>Message/@AgentName</i> .
<i>AgentVersion</i> ?	string	See <i>Message/@AgentVersion</i> .
<i>Author</i> ?	string	See <i>Message/@Author</i> .
<i>DeviceID</i> ?	NMTOKEN	See <i>Message/@DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See <i>Message/@ICSVersions</i> .
<i>ID</i> ?	ID	See <i>Response/@ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>refID</i> ?	NMTOKEN	See <i>Response/@refID</i> .
<i>ReturnCode</i> ?	integer	See <i>Response/@ReturnCode</i> .
<i>Subscribed</i> ?	boolean	See <i>Response/@Subscribed</i> .
<i>Time</i> ?	dateTime	See <i>Message/@Time</i> .
<i>Notification</i> ?	element	See <i>Response/Notification</i> .
<i>SubscriptionInfo</i> ?	element	One <i>SubscriptionInfo</i> element SHALL be returned for every Persistent Channel that was removed.

5.6 Messages Relating to Jobs and Devices

XJDF Messaging provides methods to trace the status of individual Devices and Resources and additional Job-dependent Job-tracking data. The status of a Job is described by the **Status Elements** of that Job.

Devices are uniquely identified by a *name* — that is, by the Attribute *@DeviceID* of the **Device** Resource (see Section 8.31, “Device”) — while Controllers are uniquely identified by their URL. In other words, Controllers are implicitly identified as a result of the fact that they are responding to a Message. One Controller MAY control multiple Devices. The following queries and commands are defined for status and progress tracking.

Table 5-36: Messages Relating to Jobs and Devices

Messages	Description
CommandFlushResources	Remove temporary Resources from a Device.
QueryFlushResources	
ResponseFlushResources	
SignalFlushResources	
CommandResource	Queries and/or modifies XJDF Resources that are used by a Device, such as Device settings, or by a Job. This Message can also be used to query the level of consumables in a Device.
QueryResource	
ResponseResource	
SignalResource	
CommandShutdown	Shuts down a Device.
Down	
QueryStatus	Queries the general status of a Device, Controller or Job.
ResponseStatus	
SignalStatus	
CommandWakeUp	Wakes up a Device that is in standby mode.
ResponseWakeUp	

5.6.1 FlushResources

The FlushResources Message is used to remove temporary Resources from a Device. FlushResourceParams specifies the Resources to remove.

The Command allows a Controller to Request that a Device actively Flush its resources whereas the Query or Signal allows a Device to Message that it has flushed resources to a Controller.

5.6.1.1

5.6.1.2 CommandFlushResources

The CommandFlushResources is used to remove temporary Resources from a Device. FlushResourceParams allows the specification of which Resources to remove.

Table 5-37: CommandFlushResources Message

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Command/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
RelatedCommands?	NMTOKENS	See Command/@RelatedCommands.
Time?	dateTime	See Message/@Time.
TransactionID?	NMTOKEN	See Command/@TransactionID.
FlushResourceParams?	element	Defines the Resources to be removed.

5.6.1.3 QueryFlushResources

The `QueryFlushResources` is used to query whether temporary Resources have been removed by a Device. `FlushResourceParams` allows the specification of which Resources were removed.

Table 5-38: QueryFlushResources Message

Name	Data Type	Description
<code>AgentName?</code>	string	See <code>Message/@AgentName</code> .
<code>AgentVersion?</code>	string	See <code>Message/@AgentVersion</code> .
<code>Author?</code>	string	See <code>Message/@Author</code> .
<code>DeviceID?</code>	NMTOKEN	See <code>Message/@DeviceID</code> .
<code>ICSVersions?</code>	NMTOKENS	See <code>Message/@ICSVersions</code> .
<code>ID</code>	ID	See <code>Query/@ID</code> .
<code>PersonalID?</code>	NMTOKEN	See <code>Message/@PersonalID</code> .
<code>Time?</code>	dateTime	See <code>Message/@Time</code> .
<code>Subscription?</code>	element	See <code>Query/Subscription</code> .
<code>FlushResourceParams?</code>	element	Defines the Resources to be removed.

5.6.1.3.1 Element: FlushResourceParams

Table 5-39: FlushResourceParams Element

Name	Data Type	Description
<code>FlushPolicy?</code>	enumeration	<p>Policy that defines how much of the <code>QueueEntry</code> Resources is requested to be flushed.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <code>Complete</code> – Remove all temporary Resources belonging to the selected <code>QueueEntry</code> including global resources that MAY be used by other <code>QueueEntry</code> elements. <code>QueueEntry</code> – The local Resources belonging to the selected <code>QueueEntry</code> are completely removed and no longer available — the default. <code>Intermediate</code> – Remove any intermediate Resources that belong to the <code>QueueEntry</code> (e.g., intermediate raster files in a combined RIP and Image-Setting Process), and retain the original Input Resources. A <code>PipeControl</code> Message with <code>@Operation = "Pull"</code> is still possible after executing <code>FlushResources</code> with <code>@FlushPolicy = "Intermediate"</code>.
<code>QueueFilter?</code>	element	Defines a <code>QueueFilter</code> that specifies the <code>QueueEntry</code> Elements to which the Resources to be removed belong. If not specified, all temporary resources on the Device are completely flushed according to the value of <code>@FlushPolicy</code> .

5.6.1.4 ResponseFlushResources

Table 5-40: ResponseFlushResources Message (Sheet 1 of 2)

Name	Data Type	Description
<code>AgentName?</code>	string	See <code>Message/@AgentName</code> .

Table 5-40: ResponseFlushResources Message (Sheet 2 of 2)

Name	Data Type	Description
<i>AgentVersion</i> ?	string	See Message/@AgentVersion.
<i>Author</i> ?	string	See Message/@Author.
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID.
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ICSVersions.
<i>ID</i> ?	ID	See Response/@ID.
<i>PersonalID</i> ?	NMTOKEN	See Message/@PersonalID.
<i>refID</i> ?	NMTOKEN	See Response/@refID.
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode.
<i>Subscribed</i> ?	boolean	See Response/@Subscribed.
<i>Time</i> ?	dateTime	See Message/@Time.
<i>FlushedResources</i> ?	element	This Element is a placeholder for future use.
<i>Notification</i> ?	element	See Response/Notification.

5.6.1.4.1 Element: FlushedResources

Table 5-41: FlushedResources Element

Name	Data Type	Description

5.6.1.5 SignalFlushResources

Table 5-42: SignalFlushResources Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@AgentName.
<i>AgentVersion</i> ?	string	See Message/@AgentVersion.
<i>Author</i> ?	string	See Message/@Author.
<i>ChannelMode</i> ?	enumeration	See Signal/@ChannelMode.
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID.
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ICSVersions.
<i>ID</i> ?	ID	See Signal/@ID.
<i>PersonalID</i> ?	NMTOKEN	See Message/@PersonalID.
<i>refID</i> ?	NMTOKEN	See Signal/@refID.
<i>Time</i> ?	dateTime	See Message/@Time.
<i>FlushedResources</i> ?	element	This Element is a placeholder for future use.

5.6.1.6

5.6.2 Resource

The **Resource** Message can be a **Command** Message or a **Query** Message to modify or to query **XJDF** Resources. In both cases (query and command), it is possible to address either global Device Resources, such as Device settings or Job-specific Resources. The Query Message retrieves information about the Resources without modifying them, while the Command Message modifies those settings within the Resource that are specified. Settings that are not specified remain unchanged.

5.6.2.1 QueryResource

The **QueryResource** Message can be made selective by specifying a **ResourceQuParams** Element. The presence of the **@JobID** Attribute determines whether global Device Resources or Job-related Resources are returned. If no **ResourceQuParams** Element is specified, only the global Device Resources are returned.

The query's Response Message returns a list of **ResourceInfo** Elements that contains the queried information concerning the Resources. If the list is empty because the selective query parameters of the **ResourceQuParams** lead to a null selection of the known Device/Job Resources, then the **@ReturnCode** is 103 (**@JobID** unknown), 104 (**@JobPartID** unknown) or 108 (empty list) and SHOULD be flagged as a warning with **Notification[@Class = "Warning" and @Type = "Error"]**.

Table 5-43: QueryResource Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@AgentName .
<i>AgentVersion</i> ?	string	See Message/@AgentVersion .
<i>Author</i> ?	string	See Message/@Author .
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID .
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ICSVersions .
<i>ID</i>	ID	See Query/@ID .
<i>PersonalID</i> ?	NMTOKEN	See Message/@PersonalID .
<i>Time</i> ?	dateTime	See Message/@Time .
<i>Subscription</i> ?	element	See Query/Subscription .
<i>ResourceQuParams</i> ?	element	Specifies the Resources queried.

5.6.2.1.1 Element: ResourceQuParams

Table 5-44: ResourceQuParams Element (Sheet 1 of 2)

Name	Data Type	Description
<i>ExternalID</i> ?	NMTOKEN	<i>@ExternalID</i> of the Resource that is queried.
<i>JobID</i> ?	NMTOKEN	<i>@JobID</i> of the JDF Node for which resource information is being queried. If no <i>@JobID</i> is specified, the request applies to the currently running Job or global resources, depending on the value of <i>@Context</i> . <i>@JobID</i> SHALL NOT be specified if QueryResource/Subscription is present.
<i>JobPartID</i> ?	NMTOKEN	<i>@JobPartID</i> of the JDF Node for which resource information is being queried. If no <i>@JobPartID</i> is specified, all resources related to <i>@JobID</i> are queried. <i>@JobPartID</i> SHALL NOT be specified if <i>@JobID</i> is absent. <i>@JobPartID</i> SHALL NOT be specified if QueryResource/Subscription is present.

Table 5-44: ResourceQuParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>QueueEntryID</i> ?	NMTOKEN	@QueueEntryID of the Job that is currently being executed. If @QueueEntryID is specified, @JobID, @JobPartID and Part SHALL be ignored. If none of @JobID, @JobPartID, Part or @QueueEntryID are specified, ResourceQuParams applies to all Jobs. @QueueEntryID SHALL NOT be specified if QueryResource/Subscription is present.
<i>ResourceDetails</i> ?	enumeration	Refines the level of information provided about the Resources. Allowed values are: <i>Brief</i> – Provides appropriate ID information specific to the type of Resource and @DescriptiveName Attributes only. For example, @ExternalID would be included for Resource Elements that represent consumables, @PersonalID for Contact Resources that represent employees. <i>Full</i> – Provides all of the attributes of the resources.
<i>ResourceName</i> ?	NMTOKENS	Name of the Resource(s) being queried. Values include those from: Section 8, “Resources”.
<i>Scope</i> ?	Scope	Specifies whether the query refers to a complete list of all potential Resources or to the currently loaded Resources or the Resources related to a specific job.

Example 5-7: Resource Query about Paper

The following is an example of a press system sending a Resource Query to an MIS to get information on all paper known by the MIS:

TBD 2.x Example.

```
<Query ID="M170" Type="Resource" xsi:type="QueryResource" >
  <ResourceQuParams ResourceDetails="Brief" LotDetails="Full" Scope="Allowed"/>
</Query>
```

Example 5-8: Resource Response about Paper

The following is an example of a Resource Response to the previous Resource Query

TBD 2.x Example.

```
<Response ID="M1001" Type="Resource" xsi:type="ResponseResource" refID="M170">
  <ResourceInfo LotControlled="false">
    <Media ID="R01" Class="Consumable" Status="Available"
      ProductID="9902-1" DescriptiveName="60 lb #3 Gloss Book"/>
  </ResourceInfo>
  <ResourceInfo LotControlled="true">
    <Media ID="R01" Class="Consumable" Status="Available"
      ProductID="9903-1" DescriptiveName="80 lb #3 C1S Cover"/>
    <Lot LotID="LN8845739CN7787399-03"/>
    <Lot LotID="LN8845739CN7787399-04"/>
    <Lot LotID="LN8845739CN7787399-06"/>
    <Lot LotID="LN8845739CN7787399-10"/>
  </ResourceInfo>
  <!-- ... -->
  <ResourceInfo LotControlled="false">
    <Media ID="R01" Class="Consumable" Status="Available"
      ProductID="9989-5" DescriptiveName="110 lb #1 Coated Cover"/>
```

```
</ResourceInfo>
</Response>
```

Example 5-9: Resource Query about Employees

The following is an example of a press system sending a Resource Query to an MIS to get a list of all known employees in the MIS:

TBD 2.x Example.

```
<Query ID="M170" Type="Resource" xsi:type="QueryResource">
    <ResourceQuParams resourceName="Employee" ResourceDetails="Brief"/>
</Query>
```

Example 5-10: Resource Response about Employees

The following is an example of a Resource Response to the previous Resource Query

TBD 2.x Example.

```
<Response ID="M1001" Type="Resource" xsi:type="ResponseResource"
    refID="M170">
    <ResourceInfo>
        <Employee ID="E01" Class="Implementation" Status="Available"
            PersonalID="1034" DescriptiveName="John Allen"/>
    </ResourceInfo>
    <ResourceInfo>
        <Employee ID="E02" Class="Implementation" Status="Available"
            PersonalID="1057" DescriptiveName="Sally Brown"/>
    </ResourceInfo>
    <ResourceInfo>
        <Employee ID="E03" Class="Implementation" Status="Available"
            PersonalID="2105" DescriptiveName="Mike Davison"/>
    </ResourceInfo>
    <!-- ... -->
    <ResourceInfo>
        <Employee ID="E04" Class="Implementation" Status="Available"
            PersonalID="6410" DescriptiveName="Will Smith"/>
    </ResourceInfo>
</Response>
```

Example 5-11: Resource Signal about Consumed Resources

The following is an example of a Resource Signal used to report the inventory identification of the Resources that were used:

TBD 2.x Example.

```
<Signal ID="P172" Type="Resource" xsi:type="SignalResource">
    <ResourceQuParams JobID="34028" JobPartID="_F05A84BD"/>
    <ResourceInfo>
        <Media PartIDKeys="SheetName" ID="RI007" Class="Consumable"
            ProductID="3002" Brand="Roll Stock"
            Dimension="2520 8640000" MediaType="Paper">
            <Media SheetName="1"/>
            <Media SheetName="2"/>
        </Media>
        <AmountPool>
            <PartAmount ActualAmount="9700">
                <Part SheetName="1"/>
                <Lot ActualAmount="4850" Consumption="Full"
                    LotID="LN1040788312RN20050917-04"/>
            </PartAmount>
        </AmountPool>
    </ResourceInfo>
</Signal>
```

```

<Lot ActualAmount="4850" Consumption="Partial"
      LotID="LN1040788339RN20050919-01"/>
</PartAmount>
<PartAmount ActualAmount="5027">
  <Part SheetName="2"/>
  <Lot ActualAmount="5027" Consumption="Partial"
       LotID="LN1040788319RN20050917-04"/>
</PartAmount>
</AmountPool>
</ResourceInfo>
</Signal>

```

5.6.2.2 CommandResource

The Resource Command Message is used to modify or create global Device resources such as media catalogs or lists of known machine operators. It can be made selective by specifying the OPTIONAL Attributes in the ResourceCmdParams Element.

If it is not successful, the value of *@ReturnCode* SHALL be set to "140" (Resource Command rejected). If the data could only be partially updated ResponseResource*@ReturnCode* SHALL be "141" (Resource Command partially rejected).

If the value of *@ReturnCode* is larger than "0", the Controller that issued the command SHOULD submit a global Resource Query and evaluate the returned ResourceInfo elements in order to find the setting that could not be applied.

Table 5-45: CommandResource Message

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Command/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
RelatedCommands?	NMTOKENS	See Command/@RelatedCommands.
Time?	dateTime	See Message/@Time.
TransactionID?	NMTOKEN	See Command/@TransactionID.
ResourceCmdParams?	element	Specifies the Resources to be modified.

5.6.2.2.1 Element: ResourceCmdParams

Table 5-46: ResourceCmdParams Element (Sheet 1 of 2)

Name	Data Type	Description
ExternalID?	NMTOKEN	@ExternalID of the Resource that is updated.
ResourceName?	NMTOKEN	Name of the Resource whose production amount will be modified. Values include those from: Chapter 8, “Resources”.

Table 5-46: ResourceCmdParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>UpdateMethod</i> ?	enumeration	<p>Method how the Resource is updated.</p> <p>Attributes that are required to correctly identify the Resource SHALL be specified, even if <i>@UpdateMethod</i> = "Remove" or <i>@UpdateMethod</i> = "Incremental". These Attributes include <i>@ExternalID</i>, <i>@PartIDKeys</i>, and any Partition Keys.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Complete</i> – The Resource Partitions defined by <i>Part</i> are completely overwritten by <i>Resource</i> in this Message. <i>Incremental</i> – The Resource Partitions defined by <i>Part</i> are incrementally updated by the values that are explicitly set in <i>Resource</i> in this Message. <i>Remove</i> – The Resources or Resource Partitions are removed.
<i>ResourceSet</i> ?	element	<i>ResourceSet</i> lists the Resources that are uploaded to the Device. They replace the original Resources in the Device according to the policy specified in <i>@UpdateMethod</i> .

Example 5-12: Resource Command: Single Resource is Available

The following is an example for specifying that the Cyan, Front plate of *Sheet2*, Signature 1 has become available

TBD 2.x Example.

```
<Command ID="C1" Type="Resource" xsi:type="CommandResource">
  <ResourceCmdParams JobID="MakeBrochure 012" ResourceID="ExposedMediaID"
    Status="Available">
    <Part SignatureName="Sig1" SheetName="Sheet2" Side="Front"
      Separation="Cyan"/>
  </ResourceCmdParams>
</Command>
```

Example 5-13: Resource Command: Multiple Resources are Available

The following is an example for specifying that the Black, Front plate of *Sheet2*, Signature 1 has become available and is also the last plate of Sheet 2.

TBD 2.x Example.

```
<Command ID="C2" Type="Resource" xsi:type="CommandResource">
  <ResourceCmdParams JobID="MakeBrochure 012" ResourceID="ExposedMediaID"
    Status="Available">
    <Part SignatureName="Sig1" SheetName="Sheet2" Side="Front"
      Separation="Black"/>
    <!-- the entire front of Sheet2 is also available -->
    <Part SignatureName="Sig1" SheetName="Sheet2" Side="Front"/>
    <!-- the entire Sheet2 is also available -->
    <Part SignatureName="Sig1" SheetName="Sheet2"/>
  </ResourceCmdParams>
</Command>
```

5.6.2.3 ResponseResource

Table 5-47: ResponseResource Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See <i>Message/@AgentName</i> .
<i>AgentVersion</i> ?	string	See <i>Message/@AgentVersion</i> .
<i>Author</i> ?	string	See <i>Message/@Author</i> .
<i>DeviceID</i> ?	NMTOKEN	See <i>Message/@DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See <i>Message/@ICSVersions</i> .
<i>ID</i> ?	ID	See <i>Response/@ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See <i>Message/@PersonalID</i> .
<i>refID</i> ?	NMTOKEN	See <i>Response/@refID</i> .
<i>ReturnCode</i> ?	integer	See <i>Response/@ReturnCode</i> .
<i>Subscribed</i> ?	boolean	See <i>Response/@Subscribed</i> .
<i>Time</i> ?	dateTime	See <i>Message/@Time</i> .
<i>Notification</i> ?	element	See <i>Response/Notification</i> .
<i>ResourceInfo</i> *	element	Response to a Command: contains information about the Resources after modification. Response to a Query: contains the amount data of Resources and if requested, the Resources itself.

5.6.2.3.1 Element: ResourceInfo

Table 5-48: ResourceInfo Element (Sheet 1 of 2)

Name	Data Type	Description
<i>CommandResult</i> ?	enumeration	<p>Result of a Resource Command.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Rejected</i> – the Resource Command was not applied to this Resource. <i>Removed</i> – An existing Resource was removed completely by a Resource specified in <i>ResourceCmdParams</i>. <i>New</i> – A new Resource with the values specified in <i>ResourceCmdParams</i> was created. <i>Merged</i> – Values from the Resource in <i>ResourceCmdParams</i> were merged into an existing Resource. See the <i>ResourceInfo/Resource</i> for the merged result. <i>Replaced</i> – An existing Resource was replaced completely by a Resource specified in <i>ResourceCmdParams</i>.
<i>JobID</i> ?	NMTOKEN	@ <i>JobID</i> species the @ <i>JobID</i> of the job that this <i>ResourceInfo</i> applies to.
<i>JobPartID</i> ?	NMTOKEN	@ <i>JobPartID</i> species the @ <i>JobPartID</i> of the work step that this <i>ResourceInfo</i> applies to.
<i>QueueEntryID</i> ?	NMTOKEN	@ <i>QueueEntryID</i> species the @ <i>QueueEntryID</i> of the queue entry that this <i>ResourceInfo</i> applies to.

Table 5-48: ResourceInfo Element (Sheet 2 of 2)

Name	Data Type	Description
Scope ?	Scope	@Scope specifies the context of the resources defined in this ResourceInfo.
MISDetails ?	element	Definition how the costs for the production of the Resource are to be charged.
ResourceSet *	element	XJDF Detailed description of the Resource. If the query or command leading to this ResourceInfo contains Part Elements, the ResourceSet SHALL contain only the appropriate matching Resource elements.

Example 5-14: Resource Query for Consumables

The following is an example for retrieving settings:

TBD 2.x Example.

```
<Query ID="Q1" Type="Resource" xsi:type="QueryResource">
    <ResourceQuParams Classes="Consumable" Exact="true"/>
</Query>
```

Example 5-15: Resource Response about Consumables

The following is a possible Response Message to the Query Message above:

TBD 2.x Example.

```
<Response ID="M1" Type="Resource" xsi:type="ResponseResource" refID="Q1">
    <ResourceInfo AvailableAmount="2120" Location="Paper Tray 1">
        <Media ID="ID123" Class="Consumable" Status="Available">
            <!-- Media resource defined in JDF -->
        </Media>
    </ResourceInfo>
    <ResourceInfo AvailableAmount="0" Level="Empty" Location="Ink1" Unit="1">
        <Ink ID="ID124" Class="Consumable" Status="Available">>
            <!-- Ink description resource defined in JDF -->
        </Ink>
    </ResourceInfo>
</Response>
```

Example 5-16: Resource Command for Changing Amount

The following is an example for modifying the production amount of a specific Job to produce brochures

TBD 2.x Example.

```
<Command ID="C1" Type="Resource" xsi:type="CommandResource">
    <ResourceCmdParams JobID="MakeBrochure 012" ProductionAmount="7500"
        ResourceName="Component"/>
</Command>
```

Example 5-17: Resource Response for Changing Amount

The following is a possible response to the Resource Command Message above:

TBD 2.x Example.

```
<Response ID="M2" Type="Resource" xsi:type="ResponseResource" refID="C1">
    <ResourceInfo Amount="7500" ResourceName="Component"/>
</Response>
```

5.6.2.4 SignalResource

Table 5-49: SignalResource Message

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
ChannelMode ?	enumeration	See Signal/@ChannelMode.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Signal/@ID.
PersonalID ?	NMTOKEN	See Message/@PersonalID.
refID ?	NMTOKEN	See Signal/@refID.
Time ?	dateTime	See Message/@Time.
ResourceInfo *	element	Signal from a Command: contains information about the Resources after modification. Signal from a Query: contains the amount data of Resources and if requested, the Resources itself.

```
<Command ID="C3" Type="ResourcePull" xsi:type="CommandResourcePull">
  <ResourcePullParams QueueEntryID="AllPlates" Priority="100" ResourceID="R42">
    <Part SheetName="Sheet1" Side="Front" Separation="Yellow"/>
  </ResourcePullParams>
</Command>
```

5.6.3 ShutDown

The ShutDown Command Message shuts down a Controller or Device. A Device SHALL use the Status Message if it signals its own shutdown.

5.6.3.1 CommandShutDown

Table 5-50: CommandShutDown Message (Sheet 1 of 2)

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Command/@ID.
PersonalID ?	NMTOKEN	See Message/@PersonalID.
RelatedCommands ?	NMTOKENS	See Command/@RelatedCommands.
Time ?	dateTime	See Message/@Time.
TransactionID ?	NMTOKEN	See Command/@TransactionID.

Table 5-50: CommandShutDown Message (Sheet 2 of 2)

Name	Data Type	Description
ShutDownCmdParams	element	Defines the details of a shutdown.

5.6.3.1.1 Element: ShutDownCmdParams

Table 5-51: ShutDownCmdParams Element

Name	Data Type	Description
ShutDownType ?	enumeration	<p>Defines the Device shutdown method.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>StandBy</i> – The Device is set to standby mode. It can be restarted with a WakeUp JMF Message. <i>Full</i> – Completely shut down the Device. It is no longer accessible via XJMF after the shutdown.

5.6.3.2 ResponseShutDown

Table 5-52: ResponseShutDown Message

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName .
AgentVersion ?	string	See Message/@AgentVersion .
Author ?	string	See Message/@Author .
DeviceID ?	NMTOKEN	See Message/@DeviceID .
ICSVersions ?	NMTOKENS	See Message/@ICSVersions .
ID ?	ID	See Response/@ID .
PersonalID ?	NMTOKEN	See Message/@PersonalID .
refID ?	NMTOKEN	See Response/@refID .
ReturnCode ?	integer	See Response/@ReturnCode .
Subscribed ?	boolean	See Response/@Subscribed .
Time ?	dateTime	See Message/@Time .
DeviceInfo ?	element	Describes the Device status as anticipated after the shut-down.
Notification ?	element	See Response/Notification .

5.6.4 Status

The Status Message queries the general status of a Device or a Controller and the status of Jobs associated with this Device or Controller. No Job context is needed to issue a **Status** Message. The response contains one or more **DeviceInfo** Elements, which contain the Device specific information and which MAY contain other **JobPhase** Elements that in turn contain the Job specific information. The response MAY also provide a **Queue** Element.

Table 5-53: Status Message

Object Type	Element Name	Description
QueryTypeObj	StatusQuParams	Refines the query to include various aspects of the Device and Job states.

5.6.4.1 QueryStatus

Table 5-54: QueryStatus Message

Name	Data type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Query/@ID.
PersonalID ?	NMTOKEN	See Message/@PersonalID.
Time ?	dateTime	See Message/@Time.
StatusQuParams	element	Refines the query to include various aspects of the Device and Job states.
Subscription ?	element	See Query/Subscription.

Example 5-18: Status Signal

Example of a Status Signal for a phase switch from setup to running

TBD 2.x Example.

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1"
      SenderID="MIS master A"TimeStamp="2007-08-09T11:35:41+02:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<Signal ID="m18" Type="Status" xsi:type="SignalStatus">
  <DeviceInfo DeviceStatus="Running">
    <JobPhase JobID="jID" JobPartID="jpID"
              PhaseStartTime="2007-08-09T11:35:40+02:00" Status="Setup"/>
  </DeviceInfo>
</Signal>
<Signal ID="m19" Type="Status" xsi:type="SignalStatus">
  <DeviceInfo DeviceStatus="Running">
    <JobPhase JobID="jID" JobPartID="jpID"
              PhaseStartTime="2007-08-09T11:35:41+02:00" Status="InProgress"/>
  </DeviceInfo>
</Signal>
</JMF>
```

5.6.4.1.1 Element: StatusQuParams

The various aspects of the Device, queue and Job states are refined by the **StatusQuParams** Element. This Element contains three groups of parameters. The first group serves to refine the Device-specific status information queried. The parameters **@EmployeeInfo** and **@DeviceDetails** belong to this group. The second group serves to refine the Job specific status information. These are **@JobDetails**, **@JobID** and **@JobPartID**.

In order to focus on the status of a certain Job, the Job SHALL be uniquely identified using the **@JobID** Attribute. It might be necessary to define a Process or a part of a Job as the query target under certain circumstances, such as when a Job is processed in parallel. This is accomplished using the **@JobPartID** Attribute of the **StatusQuParams** Element. **XJDFActivity** Elements SHOULD be created and put in **DeviceInfo** and/or **JobPhase** Elements when **@JobDetails = "Brief"** and **@DeviceDetails = "Details"**.

If the specified Job or Job Part is unknown, the value of the `@ReturnCode` Attribute is 103 or 104 (for error codes, see Appendix D, “Supported Error Codes in XJMF and Notification Elements” on page 1183).

Table 5-55: StatusQuParams Element

Name	Data Type	Description
<code>DeviceDetails ?</code>	enumeration	Refines the provided status information about the Device. Allowed values are: <code>None</code> – Provide only <code>DeviceInfo/@DeviceID</code> and <code>DeviceInfo/@DeviceStatus</code> . <code>Brief</code> – Provide all available Device information except for Device Elements. Contact , <code>JobPhase</code> , <code>ModuleStatus</code> and <code>Activity</code> Elements SHALL be provided if they are known. <code>Full</code> – Provide maximum available Device information. Includes Device Elements which represent details of the Device.
<code>EmployeeInfo ?</code>	boolean	If “true”, Contact Elements are to be provided in the response. Those Elements describe the employees which are associated to the Device independent on any Job.
<code>JobDetails ?</code>	enumeration	Refines the provided status information about the Jobs associated with the Device. Each higher entry includes the values specified in the lower entries. Allowed values are: <code>None</code> – Specify only <code>@JobID</code> , <code>@JobPartID</code> and <code>@Amount</code> and/or <code>@PercentCompleted</code> . <code>Brief</code> – Provide all available status information including <code>JobPhase</code> and <code>Activity</code> Elements.
<code>JobID ?</code>	NMTOKEN	<code>@JobID</code> of the JDF Node whose status is being queried. The <code>@JobID</code> SHALL be unique within the workflow. If not specified, list all known Jobs. <code>@JobID</code> SHALL NOT be specified if <code>QueryStatus/Subscription</code> is present.
<code>JobPartID ?</code>	NMTOKEN	<code>@JobPartID</code> of the JDF Node whose status is being queried. <code>@JobPartID</code> SHALL NOT be specified if <code>QueryStatus/Subscription</code> is present.
<code>QueueEntryID ?</code>	NMTOKEN	<code>@QueueEntryID</code> of the Job that is being queried. If <code>@QueueEntryID</code> is specified, <code>@JobID</code> , <code>@JobPartID</code> and <code>Part</code> are ignored. If none of <code>@JobID</code> , <code>@JobPartID</code> , <code>Part</code> or <code>@QueueEntryID</code> are specified, <code>StatusQuParams</code> applies to all Jobs. <code>@QueueEntryID</code> SHALL NOT be specified if <code>QueryStatus/Subscription</code> is present.
<code>Part *</code>	element	Part Elements that describe the Partition of the Job whose status is queried. For details on Node Partitions, see Section 4.0.1, “Partial Processing of Nodes with Partitioned Resources” on page 184. <code>Part</code> SHALL NOT be specified if <code>QueryStatus/Subscription</code> is present.

5.6.4.2 ResponseStatus

Table 5-56: ResponseStatus Message (Sheet 1 of 2)

Name	Data Type	Description
<code>AgentName ?</code>	string	See <code>Message/@AgentName</code> .
<code>AgentVersion ?</code>	string	See <code>Message/@AgentVersion</code> .
<code>Author ?</code>	string	See <code>Message/@Author</code> .

Table 5-56: ResponseStatus Message (Sheet 2 of 2)

Name	Data Type	Description
<i>DeviceID</i> ?	NMTOKEN	See Message/@ <i>DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ <i>ICSVersions</i> .
<i>ID</i> ?	ID	See Response/@ <i>ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See Message/@ <i>PersonalID</i> .
<i>refID</i> ?	NMTOKEN	See Response/@ <i>refID</i> .
<i>ReturnCode</i> ?	integer	See Response/@ <i>ReturnCode</i> .
<i>Subscribed</i> ?	boolean	See Response/@ <i>Subscribed</i> .
<i>Time</i> ?	dateTime	See Message/@ <i>Time</i> .
<i>DeviceInfo</i> *	element	Describes the actual device Status. If multiple <i>DeviceInfo</i> Elements are specified, these describe multiple Devices. A sequential state change of an individual Device SHALL be encoded as 2 separate Signals.
<i>Notification</i> ?	element	See Response/ <i>Notification</i> .

5.6.4.2.1 Element: DeviceInfo

The Response Message returns a *DeviceInfo* Element for the queried Device.

Table 5-57: DeviceInfo Element (Sheet 1 of 3)

Name	Data Type	Description
<i>CounterUnit</i> ?	NMTOKEN	The unit of the @ <i>ProductionCounter</i> , the @ <i>TotalProductionCounter</i> and numerator unit of @ <i>Speed</i> . The default unit is the default unit defined by XJDF for the Output Resource of the Node executed by the Device. For example, in case of a Sheet-Fed printer, it is the number of Sheets; in case of a Web Printer, it is the length of printed Web in meters. Values include those from: Table 1-8, "Units Used in XJDF".
<i>DeviceID</i> ?	NMTOKEN	@ <i>DeviceID</i> of the Device that this <i>DeviceInfo</i> describes. @ <i>DeviceID</i> SHALL match Device /@ <i>DeviceID</i> if Device is specified in this <i>DeviceInfo</i> .
<i>HourCounter</i> ?	duration	The total integrated time (life time) of Device operation in hours.
<i>IdleStartTime</i> ?	dateTime	Specifies the beginning of the last phase with no <i>JobPhase</i> entries. A Device is idle when no active Jobs are being processed. Multiple phases with different status values and no active Job phases MAY be specified, for instance a maintenance phase followed by an idle phase. @ <i>IdleStartTime</i> SHALL NOT be specified if <i>JobPhase</i> Elements are present in the <i>DeviceInfo</i> or @ <i>DeviceStatus</i> != "Idle", "Down" or "Stopped".
<i>PowerOnTime</i> ?	dateTime	Date and time when the Device was switched on.
<i>ProductionCounter</i> ?	double	The current Machine production counter. This counter can be reset. Typically, it starts counting at power-on time. The reset of this counter MAY be signaled by a <i>Notification</i> [@Class="Event", @Type = "CounterReset"] Message (see Section C.3, "NotificationDetails" on page 1174).
<i>Speed</i> ?	double	The current Machine speed. @ <i>Speed</i> is defined in the same units as @ <i>ProductionCounter</i> / hour.

Table 5-57: DeviceInfo Element (Sheet 2 of 3)

Name	Data Type	Description
<i>Status</i>	enumeration	<p>@Status describes the overall status of the Device.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Idle</i> – No Job is being processed and the Device is accepting new Jobs. <i>Production</i> – At least one job is in a productive status on the Device (XJDF@NodeStatus = "Setup, Running, Cleanup"). <i>Offline</i> – The Device is switched off or the machine can not be accessed. <i>Stopped</i> – At least one job is in a productive status on the Device (XJDF@NodeStatus = "Setup, Running, Cleanup") and the Device has been stopped and needs attention. This status indicates some kind of break as long as execution has not been aborted. <i>NonProductive</i> – The device is not doing productive work but rather doing something like maintenance or running a test job.
<i>StatusDetails</i> ?	string	<p>String that defines the Device state more specifically.</p> <p>Values include those from: Section C.1, “StatusDetails Supported Strings” on page 1165.</p>
<i>TotalProductionCounter</i> ?	double	The current total Machine production counter since the Machine was produced.
<i>Activity</i> *	element	Device and Operator activities that are related to the Device and are unrelated to a specific job.
<i>Contact</i> *	element	Contact Resources that describe which employees are currently working at the Device.
<i>Device</i> ?	element	A Device Resource that describes details of the Device. .
<i>FileSpec</i> (CurrentSchema) ?	element	Reference to an XML schema in XSD format [XMLSchema] that describes the present limitations of the device that can be used without operator intervention. The referenced XML schema SHALL use the xjdf namespace to describe elements and attributes that are defined in the xjdf namespace.
<i>FileSpec</i> (Schema) ?	element	Reference to an XML schema in XSD format [XMLSchema] that describes the global limitations of the device including those that can only be used with operator intervention. The referenced XML schema SHALL use the xjdf namespace to describe elements and attributes that are defined in the xjdf namespace.
<i>JobPhase</i> *	element	<p>Describes the actual status of Jobs in the Device. All Jobs that are active on the Device SHALL be specified. Supplying no JobPhase specifies that no Job is currently active on the Device.</p> <p>Multiple JobPhase Elements specify that multiple Job phases are active simultaneously on the Device.</p> <p>For details on using JobPhase Elements, see Table 5-59, “JobPhase Element” on page 249.</p>

Table 5-57: DeviceInfo Element (Sheet 3 of 3)

Name	Data Type	Description
ModuleStatus *	element	Status of individual modules that are in use independent of a Job. ModuleStatus SHALL not be specified for modules that are specified in JobPhase/ModuleStatus. For details on using ModuleStatus Elements, see Table 5-60, “ModuleStatus Element” on page 250.

5.6.4.2.2 Activity

Activity Elements allow tracking of device and operator tasks.

Table 5-58: Activity Element

Name	Data Type	Description
ActivityID ?	NMTOKEN	ID of the Activity being performed. This ID is unique, site specific and internal to the MIS.
ActivityName ?	string	Name of the Activity being performed.
PersonalID ?	NMTOKEN	MIS identifier of the employee that performs the activity.
StartTime ?	dateTime	Date and time that the Employee started the Activity. This value MAY remain the same in multiple messages.

5.6.4.3 Element: JobPhase

A Status Response Message MAY provide JobPhase Elements. The JobPhase Element represents the actual state of a Job. Any Amounts specified in JobPhase are cumulated amounts since *@StartTime*. The main difference between a JobPhase Element within an XJMF message and a AuditStatus Audit Element is that a JobPhase Message Element reflects a snapshot of the current Job status whereas the AuditStatus Audit Element reflects a time span bordered by two (sub-) status transitions.

If Part Elements are specified, all Attributes in JobPhase apply only to the specified parts.

Table 5-59: JobPhase Element (Sheet 1 of 2)

Name	Data Type	Description
Amount ?	double	Sum of actual <i>@Amount</i> that the Node defined in this JobPhase produced since <i>@StartTime</i> . If <i>@Waste</i> is also specified, the value is without waste. The unit is specified in the <i>@CounterUnit</i> Attribute of the parent Element DeviceInfo.
CostCenterID ?	string	The cost center that the Job is currently being charged to. Defaults to the cost center specified in the DeviceInfo Element.
Deadline ?	enumeration	Scheduling state of the Job. Allowed values are: <i>InTime</i> – The Job or Job Part will probably not miss the deadline. <i>Warning</i> – The Job or Job Part could miss the deadline. <i>Late</i> – The Job or Job Part will miss the deadline. Note: for more details on scheduling, see NodeInfo .
JobID ?	NMTOKEN	<i>@JobID</i> of the JDF Node that is executing.
JobPartID ?	NMTOKEN	<i>@JobPartID</i> of the JDF Node that is executing.
PercentCompleted ?	double	Node processing progress in percent (%) completed.

Table 5-59: JobPhase Element (Sheet 2 of 2)

Name	Data Type	Description
<i>QueueEntryID</i> ?	NMTOKEN	If the Job was submitted to a Queue and the @ <i>QueueEntryID</i> is known, this Attribute SHOULD be provided.
<i>RestTime</i> ?	duration	Estimated duration of time to finishing processing this Node.
<i>SpawnID</i> ?	NMTOKEN	@ <i>SpawnID</i> allows distinguishing multiple spawned Jobs with the same @ <i>JobID</i> .
<i>Speed</i> ?	double	The current Job speed. @ <i>Speed</i> is defined in the same units as @ <i>ProductionCounter</i> / hour. Defaults to the speed specified in the DeviceInfo Element.
<i>StartTime</i> ?	dateTime	Time when execution of the Node that is described by this JobPhase has been started, defined by the transition of NodeInfo/@NodeStatus from "Waiting" or "Ready" to any active value.
<i>Status</i>	enumeration	The status of the JDF Node. Allowed values are from: NodeInfo/@NodeStatus (Table 8.67, "NodeInfo" on page 709 and Table 8-148, "NodeStatus Attribute Values" on page 710).
<i>StatusDetails</i> ?	string	Machine readable description that defines the Job state more specifically. Values include those from: Section C.1, "StatusDetails Supported Strings" on page 1165.
<i>Waste</i> ?	double	Total @ <i>Amount</i> of waste that the Node defined in this JobPhase produced since @ <i>StartTime</i> . The unit is specified in the @ <i>CounterUnit</i> Attribute of the parent Element DeviceInfo.
<i>Activity</i> *	element	Device and Operator activities that are related to a specific job or job phase.
<i>MISDetails</i> ?	element	Definition how the costs for this JobPhase are to be charged.
<i>ModuleStatus</i> *	element	Status of individual modules that are used to execute this JobPhase. ModuleStatus SHALL NOT be specified for modules that are specified in DeviceInfo/ModuleStatus. For details on using ModuleStatus Elements, see Table 5-60, "ModuleStatus Element" on page 250.
<i>Part</i> *	element	Describes which parts of a Job are currently being processed. For details on Node Partitions, see Section 4.0.1, "Partial Processing of Nodes with Partitioned Resources" on page 184.

5.6.4.4 Element: ModuleStatus

The ModuleStatus Element restricts the scope of a JobPhase or DeviceInfo Element to apply only to the device modules that are selected by the list of ModuleStatus Elements.

Table 5-60: ModuleStatus Element (Sheet 1 of 2)

Name	Data Type	Description
<i>ModuleID</i>	NMTOKEN	@ <i>ModuleID</i> of the Module that ModuleStatus refers to.

Table 5-60: ModuleStatus Element (Sheet 2 of 2)

Name	Data Type	Description
<i>ModuleType</i> ?	NMTOKEN	<p>Module description</p> <p>Values include those from: Section C.2, “ModuleType Supported Strings” on page 1170.</p> <p>Note: the allowed values depend on the type of Device. Each type of Device has a separate table of values.</p>

Example 5-19: Status Response to Query

The following is an example of a Response Message to a Status Query Message. The Device in this example holds one Job and executes another Job that is currently printed duplex (each side) on four-color modules for the front and three-color modules for the back, with one idle:

TBD 2.x Example.

```
<Response ID="M1" refID="Q1" Type="Status" xsi:type="ResponseStatus">
  <DeviceInfo DeviceStatus="Running" StatusDetails="Waste">
    <JobPhase Amount="2560" DeadLine="InTime" JobID="678" JobPartID="01"
      PercentCompleted="52" QueueEntryID="Job-05" Status="InProgress"
      StatusDetails="Waste"/>
    <JobPhase Amount="0" DeadLine="Warning" JobID="679" JobPartID="01"
      PercentCompleted="0" QueueEntryID="Job-06" Status="Ready"/>
    <ModuleStatus ModuleIndex="0~3 6~8" ModuleType="PrintModule"
      DeviceStatus="Running"/>
    <ModuleStatus ModuleIndex="4" ModuleType="PrintModule" DeviceStatus="Idle"/>
    <ModuleStatus ModuleIndex="5" ModuleType="PerfectingModule"
      DeviceStatus="Running"/>
  </DeviceInfo>
</Response>
```

5.6.4.5 SignalStatus**Table 5-61: SignalStatus Message**

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@AgentName.
<i>AgentVersion</i> ?	string	See Message/@AgentVersion.
<i>Author</i> ?	string	See Message/@Author.
<i>ChannelMode</i> ?	enumeration	See Signal/@ChannelMode.
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID.
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ICSVersions.
<i>ID</i>	ID	See Signal/@ID.
<i>PersonalID</i> ?	NMTOKEN	See Message/@PersonalID.
<i>refID</i> ?	NMTOKEN	See Signal/@refID.
<i>Time</i> ?	dateTime	See Message/@Time.
<i>DeviceInfo</i> +	element	Describes the actual device Status. If multiple DeviceInfo Elements are specified, these describe multiple Devices. A sequential state change of an individual Device SHALL be encoded as 2 separate Signals.

5.6.5

```

<Command ID="ID1" Type="UpdateJDF" xsi:type="CommandUpdateJDF">
  <UpdateJDFCmdParams ParentJobID="ID100" ParentJobPartID="ID112">
    <CreateLink JobID="ID100" JobPartID="ID111">
      <MediaLink Usage="Input" rRef="link001111"/>
    </CreateLink>
    <CreateResource JobID="100" JobPartID="110">
      <Component rRef="link001112"/>
    </CreateResource>
    <RemoveLink JobID="100" JobPartID="111">
      <MediaLink Usage="Input" rRef="link001113"/>
    </RemoveLink>
    <MoveResource JobID="100" JobPartID="101" ResourceID="link000004"/>
    <JDF JobPartID="200" Type="Cutting">
      <AuditPool>
        <Created AgentName="MIS"TimeStamp="2005-06-02T09:01:45+01:00"
          AgentVersion="1.0"/>
      </AuditPool>
      <ResourcePool>
        <Component ID="link000002" Class="Quantity" Status="Available"
          ComponentType="Sheet"/>
        <CuttingParams ID="link000007" Class="Parameter" Status="Available"/>
      </ResourcePool>
      <ResourceLinkPool>
        <ComponentLink Usage="Output" rRef="link000002"/>
        <CuttingParamsLink Usage="Input" rRef="link000007"/>
      </ResourceLinkPool>
    </JDF>
  </UpdateJDFCmdParams>
</Command>

```

5.6.6 WakeUp

The WakeUp Command Message activates a Controller or Device that has been in stand-by mode. All QueueEntry elements SHALL have an *@Activation = "Hold"* and SHALL be explicitly resumed with a ModifyQueueEntry with a *"Resume"* operation. A Device SHALL use the Status Message to signal its own awakening.

5.6.6.1 CommandWakeUp

Table 5-62: CommandWakeUp Message (Sheet 1 of 2)

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Command/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
RelatedCommands?	NMTOKENS	See Command/@RelatedCommands.
Time?	dateTime	See Message/@Time.

Table 5-62: CommandWakeUp Message (Sheet 2 of 2)

Name	Data Type	Description
<i>TransactionID?</i>	NMTOKEN	See Command/@TransactionID .
<i>WakeUpCmdParams?</i>	element	Defines the details of the WakeUp Message.

5.6.6.1.1 Element: WakeUpCmdParams

WakeUpCmdParams is a placeholder for future use and for extensions to the WakeUp Message.

Table 5-63: WakeUpCmdParams Element

Name	Data Type	Description
—	—	—

5.6.6.2 ResponseWakeUp

Table 5-64: ResponseWakeUp Message

Name	Data Type	Description
<i>AgentName?</i>	string	See Message/@AgentName .
<i>AgentVersion?</i>	string	See Message/@AgentVersion .
<i>Author?</i>	string	See Message/@Author .
<i>DeviceID?</i>	NMTOKEN	See Message/@DeviceID .
<i>ICSVersions?</i>	NMTOKENS	See Message/@ICSVersions .
<i>ID?</i>	ID	See Response/@ID .
<i>PersonalID?</i>	NMTOKEN	See Message/@PersonalID .
<i>refID?</i>	NMTOKEN	See Response/@refID .
<i>ReturnCode?</i>	integer	See Response/@ReturnCode .
<i>Subscribed?</i>	boolean	See Response/@Subscribed .
<i>Time?</i>	dateTime	See Message/@Time .
<i>DeviceInfo?</i>	element	Describes the Device status immediately after the WakeUp Message has been sent.
<i>Notification?</i>	element	See Response/Notification .

5.7 Messages for Pipe Control

XJDF Messaging provides a message to control dynamic pipes. Dynamic pipes are described in detail in Section , “Overlapping Processing Using Pipes”.

Table 5-65: Messages for Control of Dynamic Pipes

Messages	Description
CommandPipeControl	All pipe related commands are implemented using the PipeControl message.
ResponsePipeControl	

5.7.1 PipeControl

The PipeControl message modulates a flow of resources in a Pipe. The type of pipe operation is specified in [PipeParams/@Operation](#)

A Pipe describes the Resource dependency in which a Process begins to consume a Resource while it is being produced by another Process (e.g., stacking components while they are being printed) or consuming a data stream while it is being written by an upstream Process. Note that defining a Pipe Resource does not automatically set up communication between Processes. The Controllers and Devices that execute the Process SHALL still implement the protocol that defines the Pipe.

Using dynamic pipe control, a downstream Process can control the total quantity produced by an upstream Process, and/or the quantity buffered by an inter-Process transport Device (i.e., Conveyor belt). Additional description of pipes and Process communication via pipes is provided in Section , “Overlapping Processing Using Pipes”.

Dependent elements MAY contain a *@PipeProtocol* that declares the Resource to be a pipe, and identifies it in a dynamic-pipe messaging environment. A pipe that is also controlled by XJMF pipe Messages is called **dynamic pipe**. or more information about dynamic pipes, see Section 4.4.3.1, “Dynamic Pipes”.

5.7.1.1 CommandPipeControl

Table 5-66: CommandPipeControl Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@AgentName.
<i>AgentVersion</i> ?	string	See Message/@AgentVersion.
<i>Author</i> ?	string	See Message/@Author.
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ICSVersions.
<i>ID</i>	ID	See Command/@ID.
<i>PersonalID</i> ?	NMTOKEN	See Message/@PersonalID.
<i>RelatedCommands</i> ?	NMTOKENS	See Command/@RelatedCommands.
<i>Time</i> ?	dateTime	See Message/@Time.
<i>TransactionID</i> ?	NMTOKEN	See Command/@TransactionID.
<i>PipeParams</i> ?	element	Details of the PipeControl message.

5.7.2 Element: PipeParams

Table 5-67: PipeParams Element (Sheet 1 of 2)

Name	Data Type	Description
<i>JobID</i>	NMTOKEN	Specifies the @JobID of the Node at the receiving end of the Message that links to the Resource specified in @PipeID.

Table 5-67: PipeParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Operation</i>	enumeration	<p><i>@Operation</i> specifies whether the flow is being pushed or pulled.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Close</i>: The PipeControl is a request to the other end of a dynamic pipe that the sender of this Message needs no further Resources or will produce no further Resources through the pipe. <i>Pause</i>: The PipeControl is a request to the other end of a dynamic pipe that the sender of this Message can currently not process Resources through the pipe. <i>Pull</i> – The PipeControl is a request to the Producer to create resources by the Consumer of the resources. <i>Push</i> – The PipeControl is a notification to the Consumer that resources have been created by the Producer of the resources.
<i>PipeID</i>	NMTOKEN	Pipe ID of the XJDF ResourceSet that defines the dynamic pipe. <i>@PipeID</i> SHALL be unique in the scope of the job that is selected by <i>@JobID</i> .
<i>Status ?</i>	enumeration	<p>Process status after the request.</p> <p>Allowed values are from: <i>NodeInfo/@NodeStatus</i> (Table 8.67, “NodeInfo” on page 709 and Table 8-148, “NodeStatus Attribute Values” on page 710).</p>
<i>UpdatedStatus ?</i>	enumeration	<p>This value represents the actual status of the pipe Resource and MAY be used by the receiving Process for Process termination control. For details see Section 4.4.5.2, “Formal Iterative Processing”.</p> <p>Allowed values are from: <i>NodeInfo/@NodeStatus</i> (Table 8.67, “NodeInfo” on page 709 and Table 8-148, “NodeStatus Attribute Values” on page 710).</p>
<i>AmountPool ?</i>	element	<p>Updated AmountPool for the pipe Resource. The AmountPool/PartAmount/Part MAY contain additional metadata related to the updated Resource.</p> <p>The ordering of the PartAmount elements in the AmountPool is relevant.</p>
<i>MISDetails ?</i>	element	Definition how the costs for the production of the Resource are to be charged.
<i>ResourceSet *</i>	element	Updated Resources to be used by the Process that receives the PipeControl command:

5.7.2.1 @Operation = Close

If *@Operation* = "Close", the PipeControl Message notifies the Process at the other end of a dynamic pipe that the sender of this Message needs no further Resources or will produce no further Resources through the pipe. The PipeControl message response with *@Operation* = "Close" is equivalent to the PipeControl message responses with *@Operation* = "Pull" and *@Operation* = "Push" described below.

If Resource/*@PipeProtocol* = "JMFPush" the producer SHALL terminate the pipe with a PipeClose Message. If Resource/*@PipeProtocol* = "JMFPull" the consumer SHALL terminate the pipe with a PipeControl message with *@Operation* = "Close".

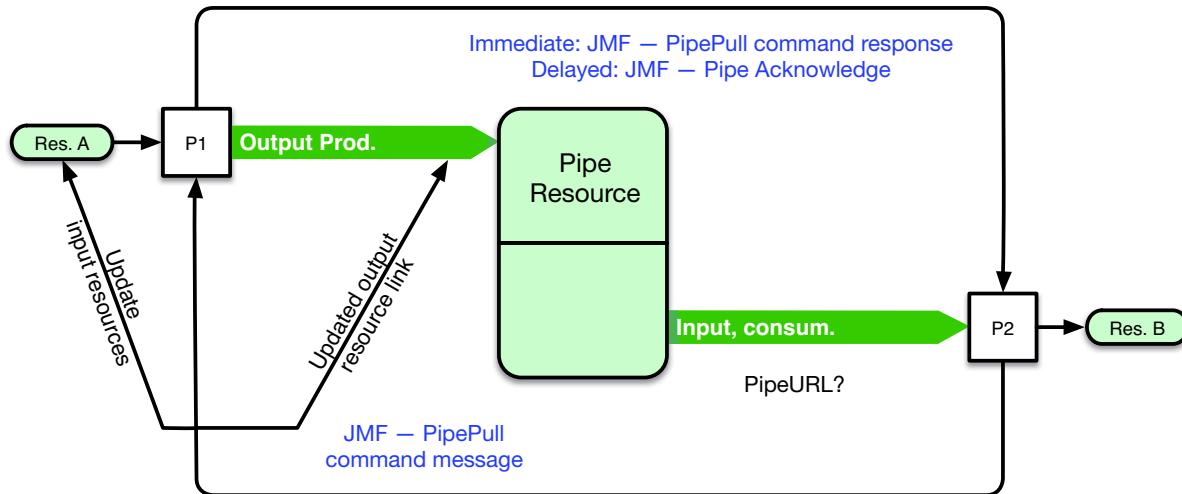
5.7.2.2 @Operation = Pull

If *@Operation* = "Pull", the PipeControl Message requests Resources that are described in an **XJDF** dynamic pipe (see Section 3.6, "Subelements of Resource" and Section , "Overlapping Processing Using Pipes"). PipePull Messages are the **XJMF** equivalent of a dynamic input Resource . Below, depicts the mode of operation of a PipeControl message with *@Operation* = "Pull".

The PipeControl message with *@Operation* = "Pull" response returns a *@ReturnCode* of 0 if the command has been accepted by the receiving Controller. If not successful the *@ReturnCode* is one of the codes presented in Section D, "Supported Error Codes in XJMF and Notification Elements". The Response Message MAY contain a **Notification Element**. The **JobPhase Element** (see Section 5.6.4, "Status") returned SHOULD provide only the *@Status* Attribute that describes the Job status of the responding Process after receiving the command.

If *Resource/@PipeProtocol* = "JMFpull", the consumer SHALL initiate the pipe with a PipeControl message with *@Operation* = "Pull".

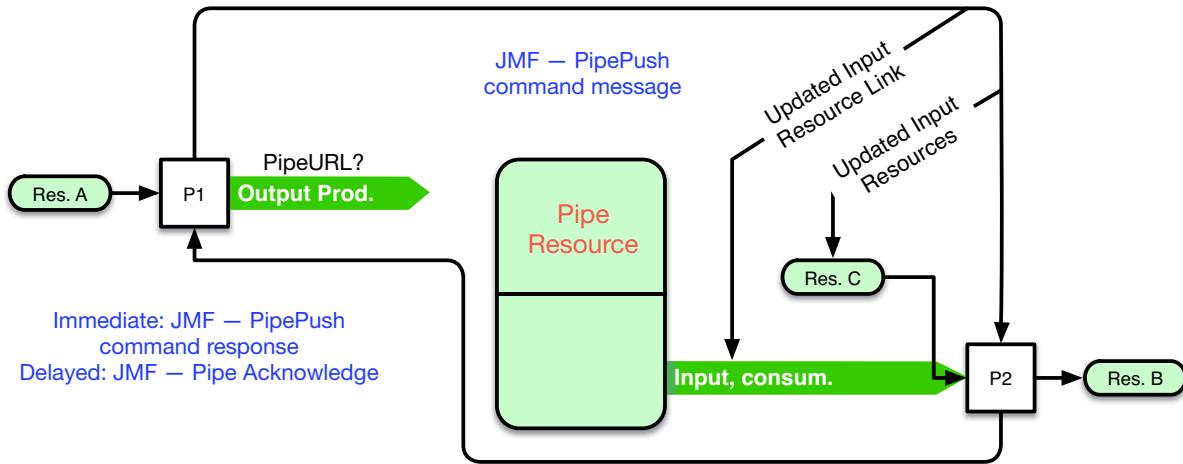
Figure 5-2: Mechanism of a PipePull Message



5.7.2.3 @Operation = Push

If *@Operation* = "Push", the PipeControl Message notifies the availability of pipe Resources that are described in an **XJDF** dynamic pipe (see Section 3.6, "Subelements of Resource" and Section , "Overlapping Processing Using Pipes"). PipePush Messages are the **XJMF** equivalent of a dynamic output Resource. The Figure 5-3 depicts the mode of operation of a PipePush Message. The PipeControl message response with *@Operation* = "Push" is equivalent to the PipeControl message response with *@Operation* = "Pull" described above.

If *Resource/@PipeProtocol* = "JMFpush", the producer SHALL initiate the pipe with a PipeControl message with *@Operation* = "Push".

Figure 5-3: Mechanism of a PipePush Message

5.7.2.4 @Operation = Pause

If *@Operation* = "Pause", the PipeControl Message pauses execution of a Process that is at the other end of a dynamic pipe.

PipePause MAY be emitted by either the consumer or the producer whenever a condition exists that requires a resynchronization.

If Resource/@*PipeProtocol* = "JMFPush", and the consumer sends a PipePause, the producer SHALL NOT send further PipePush messages until the consumer has reopened the pipe by sending a PipePull message.

If Resource/@*PipeProtocol* = "JMFPull", and the producer sends a PipePause, the consumer SHALL NOT send further PipePull messages until the producer has reopened the pipe by sending a PipePush message.

PipePause MAY be sent by the respective other end of the pipe even if the pipe is already paused. In this case the resynchronization requirements above still apply.

The PipePause Command Message response is equivalent to the PipePull Command Message response described above.

5.7.2.5 ResponsePipeControl

Table 5-68: ResponsePipeControl Message (Sheet 1 of 2)

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID?	ID	See Response/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
refID?	NMTOKEN	See Response/@refID.
ReturnCode?	integer	See Response/@ReturnCode.

Table 5-68: ResponsePipeControl Message (Sheet 2 of 2)

Name	Data Type	Description
Subscribed ?	boolean	See Response/@Subscribed.
Time ?	dateTime	See Message/@Time.
Notification ?	element	See Response/Notification.

5.8 Queue Support

In **XJMF**, a Controller or Device is assumed to have one input queue that accepts submitted Jobs. Controllers which receive submitted Jobs SHALL in turn submit these Jobs to lower level Controllers or Devices to pass the submission on. In other words, Job submission “cascades” down through Controllers until they get to the Device. Similarly, **ReturnQueueEntry** Messages “cascade” back up through each level. If a Machine supports multiple queues, it SHALL be represented by multiple logical Devices in **XJDF**. In other words, a Device SHALL NOT have more than one Queue. The simple case of a Device with no queue can be mapped to a queue with two **@Status** states: “Waiting” and “Full”. **XJMF** supports simple handling of priority queues. The following assumptions are made:

- Queues support priority. Priority SHALL only be changed for waiting Jobs. A queue MAY round priorities to the number of supported priorities, which MAY be one, indicating no priority handling.
- Priority is described by an integer from 0 to 100. Priority 100 defines a Job that SHOULD pause another Job that is in progress and commence immediately. If a Device does not support the pausing of running Jobs, it SHOULD queue a priority 100 Job before the last pending priority 100 Job.
- A Controller MAY control multiple Devices/Queues.
- Queue entries can be unambiguously identified by a **@QueueEntryID**.
- A Controller or Device MAY analyze an **XJDF** that is submitted to a queue at submission or execution time. A Queue MAY treat an **XJDF** as a closed envelope that is passed on to the Device without checking. The behavior is implementation dependent.

Some conventions used in the following sections have already been introduced in Section 5.4, “Message Template”. This affects the Message Families and the descriptive tables at the beginning of each Message section that describe the type objects related to the corresponding Message. The type objects are **QueryTypeObj**, **CommandTypeObj** and **ResponseTypeObj** (see also Figure 5-1).

5.8.1 Queue Entry ID Generation

Queue entries are accessed using a **@QueueEntryID** Attribute, which the queue’s Controller or Device generates when it receives the submitted Job, and which is returned in the **SubmitQueueEntry** Response Message. **@QueueEntryID** SHALL uniquely identify an entry within the scope of one queue. An implementation is free to choose the algorithm that generates **@QueueEntryID** values.

5.9 Messages for Queue Entry Handling

Queue-entry handling is provided so that the state of individual Jobs within a queue can be changed. Job submission, queue-entry grouping, priorities and hold / suspend / resume of entries are all supported. The individual commands are defined in the table and explained in greater detail in the sections that follow.

Table 5-69: Messages for queue entry handling

Messages	Description
CommandModifyQueueEntry ResponseModifyQueueEntryl	The QueueEntry is modified by performing: "AbortQueueEntry", "HoldQueueEntry", "RemoveQueueEntry", "ResubmitQueueEntry", "ResumeQueueEntry", "SetQueueEntryPosition", "SetQueueEntryPriority" or "SuspendQueueEntry".
CommandRequestQueueEntry ResponseRequestQueueEntryl	A new Job is requested by the Device. This Message is used to signal that a Device has processing capabilities available.
CommandResubmitQueueEntry ResponseResubmitQueueEntryl	Replaces a queue entry without affecting the entry's parameters. The command is used, for example, for late changes to a submitted XJDF.
CommandReturnQueueEntry ResponseReturnQueueEntryl	Returns a Job that had been submitted with a SubmitQueueEntry to the queue that represents the Controller that originally submitted the Job.
CommandSubmitQueueEntry ResponseSubmitQueueEntry	A Job is submitted to a queue in order to be executed.

The following table specifies the status transitions for the respective queue entry handling Messages. The error(n) indicates the ReturnCode which is returned on an illegal Status transition and the queue entry Status is unchanged. For details on error codes, see Appendix D, “Supported Error Codes in XJMF and Notification Elements” on page 1183.

The following are codes for the following table:

A: Aborted

C: Completed

H: Held

PR: PendingReturn

Rm: Removed

Rn: Running

S: Suspended

W: Waiting

number: Error that specified number (e.g., “105” means “error(105)”).

Table 5-70: Status Transitions for QueueEntry Handling Messages (Sheet 1 of 2)

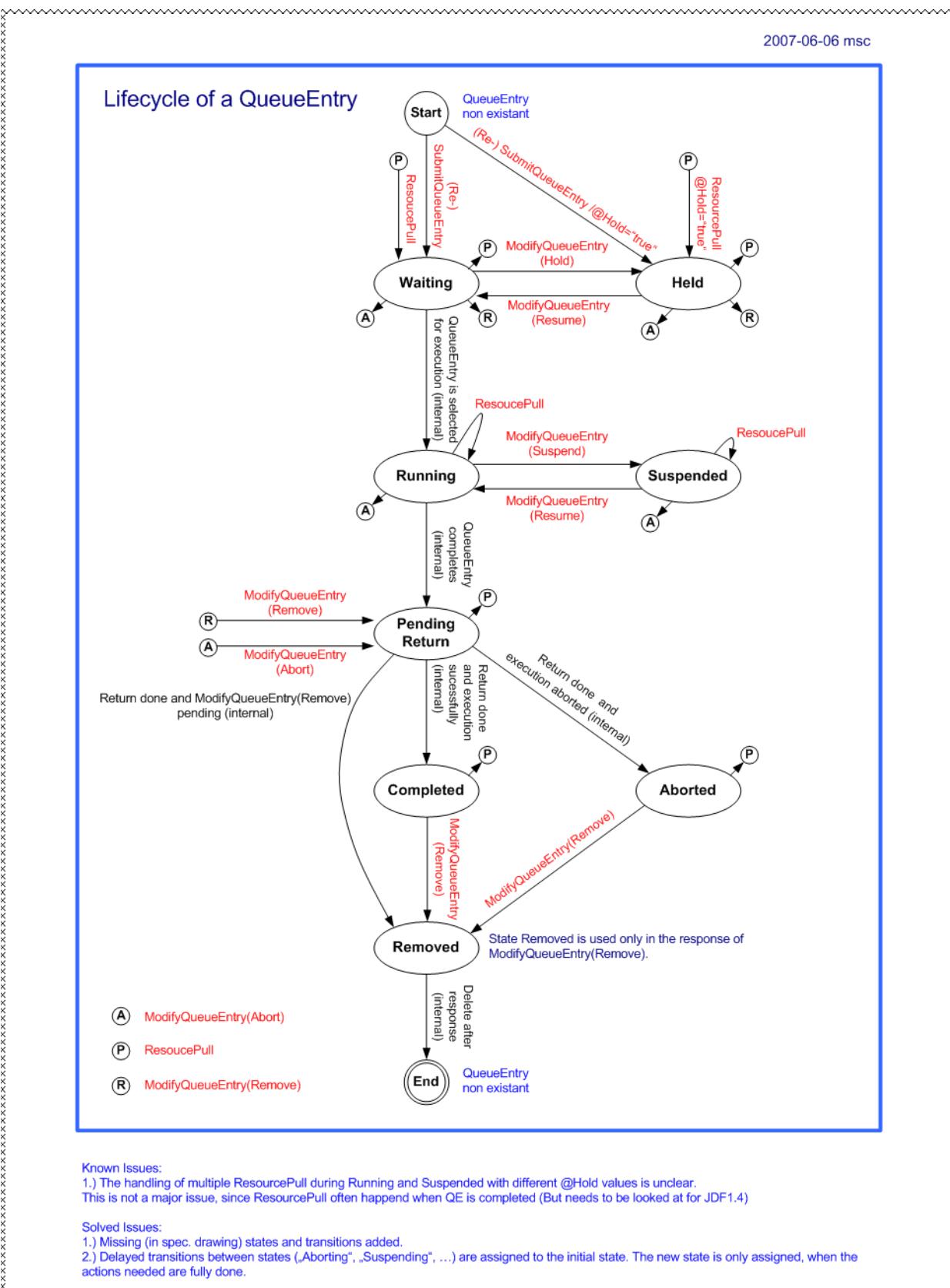
Previous Status Message type	Non- existent	W	H	Rn	S	PR	C	A
AbortQueueEntry	105	A	A	A	A	114	114	113
HoldQueueEntry	105	H	113	106	106	114	114	114
RemoveQueueEntry	105	Rm	Rm	106	106	106	Rm	Rm
ResumeQueueEntry	105	113	W	113	R/W	114	114	114
SetQueueEntryPosition	105	W	H	107	107	114	114	114
SetQueueEntryPriority	105	W	H	107	107	114	114	114
SuspendQueueEntry	105	115	115	S	113	114	114	114
RequestQueueEntry	RequestQueueEntry is emitted by the Controller of the queue and not sent to the queue. Therefore it is not applicable in this section.							

Table 5-70: Status Transitions for QueueEntry Handling Messages (Sheet 2 of 2)

Previous Status Message type	Non-existent	W	H	Rn	S	PR	C	A	
ResubmitQueueEntry	105	W	H	Rn + W + 107	S + 107	114	Rn + W + 114	Rn + W + 114	
ReturnQueueEntry	ReturnQueueEntry is emitted by the Controller of the queue and not sent to the queue. Therefore it is not applicable in this section.								
SubmitQueueEntry	W,H, Rn	A new <i>@QueueEntryID</i> is generated by the queue owner on submission. Therefore these states are not applicable.							

The following *@Status* transition diagram depicts the life cycle of a queue entry.

Figure 5-4: XJMF QueueEntry Status Transition Diagram



```

<Command ID="M009" Type="AbortQueueEntry" xsi:type="CommandAbortQueueEntry">
  <AbortQueueEntryParams>
    <QueueFilter>
      <QueueEntryDef QueueEntryID="job-0032"/>
    </QueueFilter>
  </AbortQueueEntryParams>
</Command>
<Response ID="M109" Type="AbortQueueEntry" xsi:type="ResponseAbortQueueEntry"
  refID="M009" ReturnCode="0">
</Response>

```

5.9.0.1 Element: HoldQueueEntryParams

5.9.1 ModifyQueueEntry

ModifyQueueEntry modifies the state of one or more QueueEntry elements specified by ModifyQueueEntryParams. *Operation* specifies the operation on the selected queue entries. For details, see ModifyQueueEntryParams and Table 5-70, “Status Transitions for QueueEntry Handling Messages” on page 259.

See also ResubmitQueueEntry command for modifications of the underlying XJDF without modifying the queue s.

5.9.1.1 CommandModifyQueueEntry

Table 5-72: CommandModifyQueueEntry Message

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Command/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
RelatedCommands?	NMTOKENS	See Command/@RelatedCommands.
Time?	dateTime	See Message/@Time.
TransactionID?	NMTOKEN	See Command/@TransactionID.
ModifyQueueEntryParams	element	ModifyQueueEntryParams defines the operation, the selected queue entry (or selected queue entries) and the operation to be performed.

5.9.1.1.1 Element: ModifyQueueEntryParams

Table 5-73: ModifyQueueEntryParams Element

Name	Data Type	Description
<i>NextQueueEntryID?</i>	NMTOKEN	ID of the queue entry that is to be positioned directly behind the entry. If more than one <i>QueueEntry</i> is selected, the ordering of these elements in the resulting queue is implementation dependent. Not more than one of <i>@NextQueueEntryID</i> , <i>@PrevQueueEntryID</i> <i>@Position</i> or <i>@Priority</i> SHALL be specified.
<i>Operation</i>	NMTOKEN	The operation to perform on the specified queue entries: Values include those from: Table 5-74, “Operation Attribute Values” on page 264.
<i>Position?</i>	integer	Position in the queue. “0” = pole position. Note that the position is based on the queue before modification. Thus if a queue entry is moved back in the queue, its final position is one lower than specified in <i>Position</i> . If more than one <i>QueueEntry</i> is selected, the ordering of these elements in the resulting queue is implementation dependent. Not more than one of <i>@NextQueueEntryID</i> , <i>@PrevQueueEntryID</i> , <i>@Position</i> or <i>@Priority</i> SHALL be specified.
<i>PrevQueueEntryID?</i>	NMTOKEN	ID of the queue entry that is to be positioned directly in front of the entry. If more than one <i>QueueEntry</i> is selected, the ordering of these elements in the resulting queue is implementation dependent. Not more than one of <i>@NextQueueEntryID</i> , <i>@PrevQueueEntryID</i> <i>@Position</i> or <i>@Priority</i> SHALL be specified.
<i>Priority?</i>	integer	New priority of the selected <i>QueueEntry</i> elements. Priority is a number from 0 to 100, where “0” = lowest priority and “100” = maximum priority. If more than one <i>QueueEntry</i> is selected, the ordering of these elements in the resulting queue is implementation dependent. Not more than one of <i>@NextQueueEntryID</i> , <i>@PrevQueueEntryID</i> <i>@Position</i> or <i>@Priority</i> SHALL be specified.
<i>QueueFilter</i>	element	This <i>QueueFilter</i> selects the <i>QueueEntry</i> elements to apply to

— Attribute: Operation**Table 5-74: Operation Attribute Values (Sheet 1 of 2)**

Value	Description
<i>Abort</i>	The <i>QueueEntry</i> elements selected by <i>QueueFilter</i> SHALL be aborted and remain in the <i>Queue</i> with <i>QueueEntry/@Status</i> = “Aborted”. The Audit Elements and <i>//NodeInfo//@NodeStatus</i> of the processing JDF Nodes SHALL be set to “Aborted” and the JDF Nodes SHALL be delivered to the URL as specified by <i>SubmitQueueEntry/@ReturnJMF</i> .
<i>Complete</i>	The <i>QueueEntry</i> elements selected by <i>QueueFilter</i> SHALL be stopped and remain in the <i>Queue</i> with <i>QueueEntry/@Status</i> = “Completed”. The Audit Elements and <i>NodeInfo//@NodeStatus</i> of the processing JDF Nodes SHALL be appropriately set to “Completed” and the JDF Nodes SHALL be delivered to the URL as specified by <i>SubmitQueueEntry/@ReturnJMF</i> .
<i>Hold</i>	If <i>QueueEntry/@Status</i> is “Waiting”, <i>QueueEntry/@Activation</i> SHALL be set to “Hold”. The “Hold” Operation SHALL NOT be applied to <i>QueueEntry</i> elements with a <i>@Status</i> other than “Waiting” or an <i>@Activation</i> other than “Active”. If <i>@GangPolicy</i> is other than “NoGang”, a held <i>QueueEntry</i> retains its respective gang data but SHALL NOT influence execution of other <i>QueueEntry</i> elements that are in the gang.

Table 5-74: Operation Attribute Values (Sheet 2 of 2)

Value	Description
Move	The position of the QueueEntry elements selected by QueueFilter SHALL be modified. The position of a QueueEntry SHALL NOT be modified unless <i>@Status</i> = "Waiting" or <i>@Status</i> = "Held". QueueEntry elements can either be set to a specific position within the queue or positioned next to an existing QueueEntry. QueueEntry/@Priority SHALL be set to the value of QueueEntry/@Priority of the entry that precedes it, after it has been repositioned.
Remove	The QueueEntry elements selected by QueueFilter SHALL be removed from the queue. The Remove operation does not affect QueueEntry [<i>@Status</i> = "Running" or <i>@Status</i> = "Suspended"].
Resume	If QueueEntry/@Activation = "Held", QueueEntry/@Activation SHALL be set to "Active". If QueueEntry/@Status = "Suspended", QueueEntry/@Status SHALL be set to "Running". If @GangPolicy is other than "NoGang", a resumed QueueEntry joins its respective gang.
Suspend	The QueueEntry elements selected by QueueFilter SHALL be suspended if its QueueEntry/@Status is "Running". Its QueueEntry/@Status is set to "Suspended". Whether other queue entries can be run while the queue entries remain suspended depends on implementation. The "Suspend" Operation Command Message has no effect on QueueEntry elements with a @Status other than "Running".

5.9.1.2 ResponseModifyQueueEntry

Table 5-75: ResponseModifyQueueEntry Message

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID?	ID	See Response/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
refID?	NMTOKEN	See Response/@refID.
ReturnCode?	integer	See Response/@ReturnCode.
Subscribed?	boolean	See Response/@Subscribed.
Time?	dateTime	See Message/@Time.
Notification?	element	See Response/Notification.
QueueEntry*	element	Describes the selected QueueEntry elements after the command has been executed. For the definition of the QueueEntry element, see Section 5.11, "Elements for Queues" on page 276

5.9.2 RequestQueueEntry

This command requests a new queue entry from a potential submitting Controller. The actual submission is still handled by the standard queue entry handling parameters. Note that this command is emitted from the Device that is rep-

resented by the queue to a Controller or Device and not to the queue, as is the case with most other queue handling commands.

Whereas **XJDF** generally assumes a "Push" workflow, where a controller or MIS assigns a task to a given Device, **RequestQueueEntry** allows a "Pull" workflow to be implemented, where a Device with free processing capabilities dynamically requests a new task.

5.9.2.1 CommandRequestQueueEntry

Table 5-76: CommandRequestQueueEntry Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@AgentName.
<i>AgentVersion</i> ?	string	See Message/@AgentVersion.
<i>Author</i> ?	string	See Message/@Author.
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID.
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ICSVersions.
<i>ID</i>	ID	See Command/@ID.
<i>PersonalID</i> ?	NMTOKEN	See Message/@PersonalID.
<i>RelatedCommands</i> ?	NMTOKENS	See Command/@RelatedCommands.
<i>Time</i> ?	dateTime	See Message/@Time.
<i>TransactionID</i> ?	NMTOKEN	See Command/@TransactionID.
<i>RequestQueueEntryParams</i>	element	Defines the specifics for the requested Job.

5.9.2.1.1 Element: RequestQueueEntryParams

Table 5-77: RequestQueueEntryParams Element

Name	Data Type	Description
<i>Activation</i> ?	enumeration	Specifies the Activation of the requested QueueEntry. Allowed values are from: @Activation in Table 3-1, "XJDF Node" on page 50.
<i>JobID</i> ?	NMTOKEN	@JobID of the requested QueueEntry.
<i>JobPartID</i> ?	NMTOKEN	@JobPartID of the requested QueueEntry.
<i>QueueURL</i>	URL	URL of the Queue Controller that is requesting the QueueEntry and will accept Queue manipulation Messages.
<i>Part</i> *	element	Partition parts of the requested QueueEntry.

5.9.2.2 ResponseRequestQueueEntry

The response to this message contains no element that is special for this message.

Table 5-78: ResponseRequestQueueEntry Message (Sheet 1 of 2)

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@AgentName.
<i>AgentVersion</i> ?	string	See Message/@AgentVersion.
<i>Author</i> ?	string	See Message/@Author.
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID.

Table 5-78: ResponseRequestQueueEntry Message (Sheet 2 of 2)

Name	Data Type	Description
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ <i>ICSVersions</i> .
<i>ID</i> ?	ID	See Response/@ <i>ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See Message/@ <i>PersonalID</i> .
<i>refID</i> ?	NMTOKEN	See Response/@ <i>refID</i> .
<i>ReturnCode</i> ?	integer	See Response/@ <i>ReturnCode</i> .
<i>Subscribed</i> ?	boolean	See Response/@ <i>Subscribed</i> .
<i>Time</i> ?	dateTime	See Message/@ <i>Time</i> .
<i>Notification</i> ?	element	See Response/ <i>Notification</i> .

5.9.3 ResubmitQueueEntry

A QueueEntry is resubmitted to a queue using the ResubmitQueueEntry Message. This allows late changes to be made to a QueueEntry without affecting queue parameters. Resubmission modifies the QueueEntry with XJDF information specified in ResubmissionParams/@URL. If QueueEntry/@Status is neither "Waiting" nor "Held", resubmitting a queue entry MAY fail because a Device NEED NOT implement ResubmitQueueEntry for running queue entries. See also ModifyQueueEntry command for modifications of the queue s without modifying the underlying XJDF.Job resubmission. Resubmission does not affect other queue parameters as specified. For example, resubmission does not affect queue ordering. For details, see Table 5-70, "Status Transitions for QueueEntry Handling Messages" on page 259.

5.9.3.1 CommandResubmitQueueEntry

Table 5-79: CommandResubmitQueueEntry Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@ <i>AgentName</i> .
<i>AgentVersion</i> ?	string	See Message/@ <i>AgentVersion</i> .
<i>Author</i> ?	string	See Message/@ <i>Author</i> .
<i>DeviceID</i> ?	NMTOKEN	See Message/@ <i>DeviceID</i> .
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ <i>ICSVersions</i> .
<i>ID</i>	ID	See Command/@ <i>ID</i> .
<i>PersonalID</i> ?	NMTOKEN	See Message/@ <i>PersonalID</i> .
<i>RelatedCommands</i> ?	NMTOKENS	See Command/@ <i>RelatedCommands</i> .
<i>Time</i> ?	dateTime	See Message/@ <i>Time</i> .
<i>TransactionID</i> ?	NMTOKEN	See Command/@ <i>TransactionID</i> .
<i>ResubmissionParams</i>	element	Defines the Job resubmission.

5.9.3.1.1 Element: ResubmissionParams

ResubmissionParams provides details of the QueueEntry resubmission. The value of ResubmissionParams/@*UpdateMethod* determines how the QueueEntry modification SHALL be applied.

Devices NEED NOT support @*UpdateMethod* = "Incremental" with variable XJDF/@*JobPartID*. This feature allows MIS to provide complex workflows to production workflow systems that are capable of managing multiple devices.

5.9.3.1.2 Referencing values for incremental update

The following section describes how to reference data when `ResubmissionParams/@UpdateMethod="Incremental"` or `ResubmissionParams/@UpdateMethod="Remove"`. The algorithms shown here are illustrated assuming an internal XJDF model but this is purely for illustration and no assumption is made about the actual implementation.

As a general rule, Elements and attributes within XJDF SHALL be addressed by searching Elements and Attributes with matching XPath results. Attributes with a data type of ID or IDREF SHALL be ignored when calculating XPaths because IDs MAY be regenerated dynamically and are only valid within the scope of a single XML document.

5.9.3.1.2.1 Finding the correct XJDF node to update

An XJDF node SHALL match if the values of `XJDF/@JobID` and `XJDF/@JobPartID` are both identical.

5.9.3.1.2.2 Finding the correct Resource to update

A ResourceSet SHALL match if the values of `ResourceSet/@Name`, `ResourceSet/@ProcessUsage`, `ResourceSet/@Usage`, are all identical.

Once a matching ResourceSet has been found, a child Resource SHALL match if all `Resource/Part` elements match. A Part matches if all attribute values are identical. The ordering of the `Resource/Part` is not significant.

5.9.3.1.3 Updating values

If `ResubmissionParams/@UpdateMethod="Incremental"` then all attribute values and element text SHALL be replaced with the attribute values defined in the XJDF that is referenced by `@URL`. If the ancestors of a given attribute do not exist, they SHALL be appropriately created.

5.9.3.1.4 Removing values

If `ResubmissionParams/@UpdateMethod="Remove"` then all leaf elements, i.e. elements that have no child elements, that are defined in the XJDF that is referenced by `@URL` SHALL be removed.

Table 5-80: ResubmissionParams Element

Name	Data Type	Description
<code>QueueEntryID</code>	NMTOKEN	ID of the queue entry to be replaced.
<code>UpdateMethod</code>	enumeration	<p>Method of how the <code>QueueEntry</code> SHALL be updated.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <code>Complete</code> – The <code>QueueEntry</code> SHALL be completely replaced by the XJDF that is referenced by <code>@URL</code>. <code>Incremental</code> – The <code>QueueEntry</code> SHALL be incrementally updated by the values of the XJDF that is referenced by <code>@URL</code>. If <code>XJDF/@JobPartID</code> of the referenced XJDF is identical to <code>XJDF/@JobPartID</code> of the originally submitted XJDF or an XJDF that has been resubmitted with <code>ResubmissionParams/@UpdateMethod = "Incremental"</code>, then the process step that is identified by <code>XJDF/@JobID</code> and <code>XJDF/@JobPartID</code> SHALL be modified. Otherwise a new process step SHALL be submitted to the Device. <code>Remove</code> – The <code>QueueEntry</code> SHALL be incrementally updated by removing all elements that are specified in the XJDF that is referenced by <code>@URL</code>. <code>XJDF/@JobPartID</code> of the referenced XJDF SHALL be identical to <code>XJDF/@JobPartID</code> of the originally submitted XJDF or an XJDF that has been resubmitted with <code>ResubmissionParams/@UpdateMethod = "Incremental"</code>.
<code>URL</code>	URL	Location of the XJDF to be submitted. <code>XJDF/@JobID</code> SHALL be identical to <code>XJDF/@JobID</code> of the originally submitted XJDF.

5.9.3.2 ResponseResubmitQueueEntryl

Table 5-81: ResponseResubmitQueueEntry Message

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID ?	ID	See Response/@ID.
PersonalID ?	NMTOKEN	See Message/@PersonalID.
refID ?	NMTOKEN	See Response/@refID.
ReturnCode ?	integer	See Response/@ReturnCode.
Subscribed ?	boolean	See Response/@Subscribed.
Time ?	dateTime	See Message/@Time.
Notification ?	element	See Response/Notification.

5.9.4 ReturnQueueEntry

The ReturnQueueEntry Message returns a Job that had been submitted with a SubmitQueueEntry to the Controller that originally submitted the Job. Note that this command is sent from the Device to a Controller and not from Controller to Device as is the case with most other queue handling commands.

If the XJDF has been enhanced by submitting additional process XJDFs with different XJDF/@JobPartID using the ResubmitQueueEntry Command, then only the primary XJDF SHALL be returned. The Audit elements of the process XJDFs SHALL be copied into XJDF/AuditPool of the primary XJDF. Each such Audit element SHALL contain a copy of XJDF/@JobPartID.

5.9.4.1 CommandReturnQueueEntry

Table 5-82: CommandReturnQueueEntry Message (Sheet 1 of 2)

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Command/@ID.
PersonalID ?	NMTOKEN	See Message/@PersonalID.
RelatedCommands ?	NMTOKENS	See Command/@RelatedCommands.
Time ?	dateTime	See Message/@Time.
TransactionID ?	NMTOKEN	See Command/@TransactionID.

Table 5-82: CommandReturnQueueEntry Message (Sheet 2 of 2)

Name	Data Type	Description
ReturnQueueEntryParams	element	Defines the Job being returned from Device to Controller after processing is completed or aborted.

5.9.4.1.1 Element: ReturnQueueEntryParams

The *URL* Attribute specifies the location where the **XJDF** file to be submitted can be retrieved by the Controller. The scheme of the *URL* Attribute (such as "file", "http" or "cid") defines the retrieval method to be used to retrieve the **XJDF**.

Table 5-83: ReturnQueueEntryParams Element

Name	Data Type	Description
QueueEntryID	NMTOKEN	QueueEntry/@QueueEntryID of the returned queue entry.
URL	URL	Location of the XJDF that represents the final state of the QueueEntry to be returned. URL SHALL NOT reference a directory.

5.9.4.2 ResponseReturnQueueEntry

Table 5-84: ResponseReturnQueueEntry Message

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID?	ID	See Response/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
refID?	NMTOKEN	See Response/@refID.
ReturnCode?	integer	See Response/@ReturnCode.
Subscribed?	boolean	See Response/@Subscribed.
Time?	dateTime	See Message/@Time.
Notification?	element	See Response/Notification.

5.9.5 SubmitQueueEntry

SubmitQueueEntry initially submits a QueueEntry to a Device. Modifications to a QueueEntry can be applied by using the ResubmitQueueEntry or ModifyQueueEntry command. QueueSubmissionParams provides the parameters of the submission.

ResponseSubmitQueueEntry/QueueEntry/@QueueEntryID SHALL be unique within the Device. A new @QueueEntryID SHALL be generated for each SubmitQueueEntry.

5.9.5.1 CommandSubmitQueueEntry

Table 5-85: CommandSubmitQueueEntry Message

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Command/@ID.
PersonalID ?	NMTOKEN	See Message/@PersonalID.
RelatedCommands ?	NMTOKENS	See Command/@RelatedCommands.
Time ?	dateTime	See Message/@Time.
TransactionID ?	NMTOKEN	See Command/@TransactionID.
QueueSubmissionParams	element	Defines the Job submission.

5.9.5.1.1 Element: QueueSubmissionParams

The Job submission can contain queue-ordering Attributes equivalent to those used by the "*Move*" @Operation of the ModifyQueueEntry Messages. @ReturnJMF MAY specify the location where the modified XJDF is to be sent after the Job is completed or aborted.

The @URL Attribute specifies the location where the queue Controller can retrieve the XJDF file to be submitted.

Table 5-86: QueueSubmissionParams Element (Sheet 1 of 2)

Name	Data Type	Description
Activation ?	enumeration	Activation of the submitted QueueEntry. Allowed values are from: QueueEntry/@Activation in Table 5-96, "QueueEntry Element" on page 277. Note: @Activation SHALL NOT be set to "Removed" or "PendingReturn".
GangName ?	NMTOKEN	Name of the Gang for the Job. If @GangName is specified, the QueueEntry SHOULD be executed together with other QueueEntry Elements that share a common value of @GangName. If @GangName is not known, the receiving Device MAY either return an error 131 or create the gang with @GangName on the fly.

Table 5-86: QueueSubmissionParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>GangPolicy</i> ?	enumeration	<p>Ganging policy for the QueueEntry.</p> <p>Allowed values are:</p> <p><i>Gang</i> – The Job SHALL be ganged in the gang that is specified by <i>@GangName</i> or SHALL be calculated from other properties of the submitted Job. A gang Job that MAY contain this submitted QueueEntry MAY be queued.</p> <p><i>GangAndForce</i> – The Job SHALL be ganged in the gang that is specified by <i>@GangName</i> or SHALL be calculated from other properties of the submitted Job. A gang Job that SHALL contain this submitted QueueEntry SHALL be queued.</p> <p><i>NoGang</i> – The Job SHALL NOT be ganged and <i>@GangName</i> SHALL be ignored. The Job SHALL be queued individually.</p>
<i>Hold</i> ?	boolean	If "true", the entry is submitted as with QueueEntry/ <i>@Status="Hold"</i> . If a QueueEntry is submitted with <i>@Hold = "true"</i> and <i>@GangPolicy</i> is other than " <i>NoGang</i> ", the QueueEntry retains its respective gang data but does not influence execution of other Jobs that are in the gang.
<i>NextQueueEntryID</i> ?	NMTOKEN	ID of the queue entry that is to be positioned directly behind the entry. At most one of <i>@NextQueueEntryID</i> , <i>@PrevQueueEntryID</i> or <i>@Priority</i> SHALL be specified.
<i>PrevQueueEntryID</i> ?	NMTOKEN	ID of the queue entry that is to be positioned directly in front of the entry. At most one of <i>@NextQueueEntryID</i> , <i>@PrevQueueEntryID</i> or <i>@Priority</i> SHALL be specified.
<i>Priority</i> ?	integer	<p>Number from 0 to 100, where "0" = lowest priority and "100" = maximum priority. Exactly one of <i>@NextQueueEntryID</i>, <i>@PrevQueueEntryID</i> or <i>@Priority</i> SHALL be specified.</p> <p>Note that QueueSubmissionParams/<i>@Priority</i> is not the same as NodeInfo/@Priority. QueueSubmissionParams/<i>@Priority</i> specifies the priority in the context of the Device queue whereas NodeInfo/@Priority specifies the priority of the task in general.</p> <p>QueueSubmissionParams/<i>@Priority</i> MAY be modified due to additional scheduling information (e.g., NodeInfo/@FirstStart).</p> <p>The priority from QueueSubmissionParams/<i>@Priority</i> and ModifyQueueEntryParams/<i>@Priority</i> takes precedence over NodeInfo/@JobPriority.</p>
<i>refID</i> ?	NMTOKEN	Copy of the <i>@ID</i> Attribute of the initiating RequestQueueEntry Message.
<i>ReturnJMF</i> ?	URL	Address queue where a ReturnQueueEntry Command SHALL be sent when the QueueEntry is completed or aborted.
<i>URL</i>	URL	<p>Location of the XJDF to be submitted or resubmitted. If <i>@URL</i> refers to a directory, then all files with an extension of .xjdf that reside directly in the directory SHALL be processes in lexical order. The first XJDF is referred to as the primary XJDF. See 12.4.1 Referencing Multiple XJDF in a directory.</p> <p>Note: a referenced directory MAY be inside a ZIP package. See details for zip packaging at https://www.pkware.com/support/zip-app-note/.</p>

```
<Command ID="M1" Type="SubmitQueueEntry" xsi:type="CommandSubmitQueueEntry">
  <QueueSubmissionParams URL="File:///MyNetworkShare/AnyDirectory/job1.jdf"/>
</Command>
```

URL with “http” Scheme

In this example, the queue Controller retrieves the file with a standard HTTP **get** command from a host that MAY be remote. The Job delivered as a response to the HTTP **get** command MAY be a MIME Multipart/Related entity. The HTTP server MAY retrieve a file or it MAY generate the response dynamically with a CGI script or other such tool.

Example 5-20: SubmitQueueEntry Command with “http” Scheme

TBD 2.x Example.

```
<Command ID="M2" Type="SubmitQueueEntry" xsi:type="CommandSubmitQueueEntry" >
  <QueueSubmissionParams URL="http://JobServer.JDF.COM?job1"/>
</Command>
```

5.9.5.2 ResponseSubmitQueueEntry

Table 5-87: ResponseSubmitQueueEntry Message

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID?	ID	See Response/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
refID?	NMTOKEN	See Response/@refID.
ReturnCode?	integer	See Response/@ReturnCode.
Subscribed?	boolean	See Response/@Subscribed.
Time?	dateTime	See Message/@Time.
Notification?	element	See Response/Notification.
QueueEntry?	element	Provides the queue entry of the submitted Job. QueueEntry SHALL be specified if the submission was successful and SHALL be omitted in case the submission was rejected.

5.9.5.3 Element: SuspendQueueEntryParams

5.10 Messages for Global Handling of Queues

Whereas the commands in the preceding section change the state of an individual queue entry, the commands in this section modify the state of an entire queue. Note that entries that are executing in a Device are not affected by the global queue-handling commands and SHALL be accessed individually. An individual queue can be selected by specifying the target Device in the **@DeviceID** Attribute of the JMF Root. If no **@DeviceID** is specified, the com-

mands or queries are applied to all queues that are controlled by the Controller that received the Message. The following individual Messages are defined:

Table 5-89: Messages for global handling of queues

Messages	Description
QueryQueueStatus	Returns the Queue Element that describes a queue.
ResponseQueueStatus	
SignalQueueStatus	

The following table shows the resulting status of a Queue in dependence on global queue commands "*CloseQueue*" / "*OpenQueue*" and "*HoldQueue*" / "*ResumeQueue*" as well as the load of queue and its processor. The first command pair determines the logical state of the first column "Closed" and the second of the column "Held". The Queue is held if the Queue manager doesn't send existing entries to the Queue's processor.

Table 5-90: Definition of the Queue Status Attribute Values

Closed	Held	Queue Full	Processor Full	Status
Yes	Yes	Any	Any	"Blocked"
Yes	No	Any	Any	"Closed"
No	Yes	Any	Any	"Held"
No	No	Any	No	"Waiting"
No	No	No	Yes	"Running"
No	No	Yes	Yes	"Full"

5.10.1 QueueStatus

Returns a queue description.

5.10.1.1 QueryQueueStatus

Table 5-91: QueryQueueStatus Message

Name	Data Type	Description
AgentName?	string	See Message/@AgentName.
AgentVersion?	string	See Message/@AgentVersion.
Author?	string	See Message/@Author.
DeviceID?	NMTOKEN	See Message/@DeviceID.
ICSVersions?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Query/@ID.
PersonalID?	NMTOKEN	See Message/@PersonalID.
Time?	dateTime	See Message/@Time.
QueueStatusParams	element	QueueStatusParams defines a filter for the QueueStatus Message.
Subscription?	element	See Query/Subscription.

5.10.1.1.1 Element: QueueStatusParams

Table 5-92: QueueStatusParams Element

Name	Data Type	Description
<i>PreviewUsages</i> ?	enumerations	<p>Specifies the particular kind (or kinds) of Preview Resources to return in QueueEntry/Preview. If <i>@PreviewUsages</i> is empty or not supplied, the QueueEntry Element SHALL NOT contain any Preview Resources.</p> <p>The Preview Resources returned in a QueueEntry are a subset of those in the actual QueueEntry defined by:</p> <p>QueueEntry/Preview [contains (QueueFilter/ <i>@PreviewUsages</i>, <i>@PreviewUsage</i>)]</p> <p>Allowed values are from: Preview/@PreviewUsages (Table 8-163, “Preview Resource” on page 724).</p>
<i>UpdateGranularity</i> ?	enumeration	<p>Specifies whether all or only the updated QueueEntry Elements should be included in the Queue.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>All</i> – The Queue Element describes all QueueEntry Elements. <i>ChangesOnly</i> – The Queue Element describes only those QueueEntry Elements that have new information since the last Queue Element was sent. When used in conjunction with a Signal, the Queue Element describes all Jobs on the first instance of the Signal being sent.
<i>QueueFilter</i> ?	element	Filter that filters the QueueEntry elements that SHALL be returned in the responses Queue .

5.10.1.2 ResponseQueueStatus

Table 5-93: ResponseQueueStatus Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@AgentName .
<i>AgentVersion</i> ?	string	See Message/@AgentVersion .
<i>Author</i> ?	string	See Message/@Author .
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID .
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ICSVersions .
<i>ID</i> ?	ID	See Response/@ID .
<i>PersonalID</i> ?	NMTOKEN	See Message/@PersonalID .
<i>refID</i> ?	NMTOKEN	See Response/@refID .
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode .
<i>Subscribed</i> ?	boolean	See Response/@Subscribed .
<i>Time</i> ?	dateTime	See Message/@Time .
<i>Notification</i> ?	element	See Response/Notification .
<i>Queue</i> ?	element	Describes the status of the queue.

5.10.1.3 SignalQueueStatus

Table 5-94: SignalQueueStatus Message

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
ChannelMode ?	enumeration	See Signal/@ChannelMode.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Signal/@ID.
PersonalID ?	NMTOKEN	See Message/@PersonalID.
refID ?	NMTOKEN	See Signal/@refID.
Time ?	dateTime	See Message/@Time.
Queue	element	Describes the status of the queue.

```

<Response ID="M1" Type="SubmissionMethods"
    xsi:type="ResponseSubmissionMethods" refID="Q1">
    <SubmissionMethods HotFolder="file:///MyDevice/HotFolder" Packaging="MIME"
        URLschemes="http file ftp"/>
</Response>

```

5.11 Elements for Queues

In this section Elements used by queue-handling commands are defined.

5.11.1 Queue

The Attributes in the following table are defined for Queue Message Elements. Queue Elements represent the queue of a Device including QueueEntry Elements that represent both pending and running queue entries.

Table 5-95: Queue Element

Name	Data Type	Description
DeviceID	NMTOKEN	Identifies the Device that is represented by the queue.
QueueSize ?	integer	The total number of QueueEntry Elements that are in the Queue. Note: QueueEntry[@Status = "Completed" or @Status = "Aborted"] Elements SHALL NOT be counted when computing @QueueSize. @QueueSize SHALL NOT be specified if QueueEntry elements are specified.
QueueEntry *	element	Each queue entry that was selected by QueueFilter SHALL be provided as an individual QueueEntry element. The entries SHALL be ordered in the sequence they have been or will be executed, beginning with the running entries, followed by the waiting entries, highest QueueEntry/@Priority first, which are then followed by the completed entries, sorted beginning with the youngest QueueEntry/@EndTime.

Example 5-21: Queue Element

Example of a Queue Element:

TBD 2.x Example.

```
<Queue DeviceID="Q12345" Status="Running">
  <QueueEntry JobID="111" JobPartID="1" Priority="1" QueueEntryID="111-1"
    Status="Running"/>
  <QueueEntry JobID="111" JobPartID="2" Priority="1" QueueEntryID="111-2"
    Status="Waiting"/>
  <QueueEntry JobID="112" JobPartID="1" Priority="55" QueueEntryID="112-1"
    Status="Held"/>
  <QueueEntry JobID="111" JobPartID="0" Priority="1" QueueEntryID="111-0"
    Status="Completed"/>
</Queue>
```

5.11.2 QueueEntry

Table 5-96: QueueEntry Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Activation</i> ?	enumeration	<p>Specifies the Activation of the QueueEntry. The values are ordered from least to most active.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Informative</i> – The QueueEntry is for information only. If a QueueEntry is "<i>Informative</i>", it SHALL NOT be processed. Queue entries with <i>@Activation</i> = "<i>Informative</i>" will generally be sent to an operator console for preview but are still completely under the control of an external Controller. <i>Held</i> – The QueueEntry has been held and SHALL NOT be processed until its <i>@Activation</i> is changed to "<i>Active</i>". The transition to Active MAY be triggered with a CommandModifyQueueEntry/<i>@Action</i> = "<i>Resume</i>". <i>Active</i> – The QueueEntry is active and SHALL be processed regularly. <i>PendingReturn</i> – Indicates that the QueueEntry has been processed but the has not yet been successfully returned to the respective Controller. <i>Removed</i> – The QueueEntry has been removed. This <i>@Activation</i> SHALL NOT be provided unless QueueStatusParams/<i>@UpdateGranularity</i> = "<i>ChangesOnly</i>".
<i>DeviceID</i> ?	NMTOKEN	Identification of the Device that the QueueEntry will be or was executed on. If not specified, it defaults to the default Device of the queue.
<i>EndTime</i> ?	dateTime	Time when the Job has been ended.
<i>GangName</i> ?	NMTOKEN	Name of the gang that this QueueEntry belongs to. <i>@GangName</i> SHALL be specified, if the QueueEntry is a candidate member of a gang Job.
<i>GangPolicy</i> ?	enumeration	<p>Ganging policy for the QueueEntry.</p> <p>Allowed values are from: QueueSubmissionParams/<i>@GangPolicy</i> (Table 5-86, “QueueSubmissionParams Element” on page 271).</p>
<i>JobID</i> ?	NMTOKEN	The <i>@JobID</i> of the XJDF Process.
<i>JobPartID</i> ?	NMTOKEN	The <i>@JobPartID</i> of the XJDF Process.
<i>Priority</i> ?	integer	Priority of the QueueEntry. Values are 0-100. "0" is the lowest priority, while "100" is the highest priority.

Table 5-96: QueueEntry Element (Sheet 2 of 2)

Name	Data Type	Description
<i>QueueEntryID</i>	NMTOKEN	ID of a QueueEntry. This ID SHALL be generated by the queue owner.
<i>StartTime?</i>	dateTime	Time when the Job has been started.
<i>Status</i>	enumeration	Specifies the Status of the requested QueueEntry. <i>@Status</i> SHALL be identical to the NodeInfo/@NodeStatus of the underlying XJDF. Allowed values are from: Table 8-148, “NodeStatus Attribute Values” on page 710. XJDF
<i>StatusDetails?</i>	string	<i>@StatusDetails</i> provides additional details on the status of the QueueEntry. Values include: <i>HeldForPipe</i> – When <i>@Status</i> is “PendingReturn”, Job is not returned on purpose, commands PipeControl or ModifyQueueEntry are possible <i>JobUserInputRequired</i> – When <i>@Status</i> is “Waiting” or “Running”, Job is not producible and waits for user input required to process further (e.g., missing parameters, decisions, etc.) <i>JobMissResources</i> – When <i>@Status</i> is “Waiting” or “Running”, Job waits for resources to become available to process further <i>JobReadyForStart</i> – When <i>@Status</i> is “Waiting” or “Running”, Job is ready and waits for (manual) start event to process further <i>QueuedToRun</i> – When <i>@Status</i> is “Waiting” or “Running”, Job is queued to run and waits for device to become available (idle) to process further <i>PendingReturn</i> – When <i>@Status</i> is “PendingReturn”, Job is currently returning (explicit “PendingReturn” to distinguish from devices/controllers that do not support <i>@StatusDetails</i>) <i>Running</i> – When <i>@Status</i> is “Running”, Job is processing (explicit Running to distinguish from devices/controllers that do not support <i>@StatusDetails</i>)
<i>SubmissionTime?</i>	dateTime	Time when the entry was submitted to the queue.
<i>Part*</i>	element	Describes which parts of a Job were submitted to the queue. This is a copy of the partitioning structure of the NodeInfo element of the attached XJDF.
<i>Preview*</i>	element	Any number of Preview Elements MAY be associated with a QueueEntry and used for display purposes. Preview/@PreviewUsage SHOULD be “Thumbnail” or “Viewable”.

5.11.3 QueueFilter

The QueueFilter Element defines a filter that selects QueueEntry elements in a Queue. The supplied Elements of the QueueFilter define a matching criteria that is a logical “and”. Only QueueEntry Elements that match all restrictions specified by the QueueFilter SHALL be selected.

Table 5-97: QueueFilter Element

Name	Data Type	Description
<i>GangNames</i> ?	NMTOKENS	Gang names of the QueueEntry Elements to be returned. If not specified, there is no filtering on QueueEntry/@ <i>GangName</i> .
<i>JobID</i> ?	NMTOKEN	Return only QueueEntry Elements with specified @ <i>JobID</i> . If not specified, there is no filtering on QueueEntry/@ <i>JobId</i> .
<i>JobPartID</i> ?	NMTOKEN	Return only QueueEntry Elements with specified @ <i>JobPartID</i> . If not specified, there is no filtering on QueueEntry/@ <i>JobPartID</i> .
<i>MaxEntries</i> ?	integer	Maximum number of QueueEntry Elements to provide in the Queue Element. If not specified, fill in all matching QueueEntry Elements.
<i>OlderThan</i> ?	dateTime	Only QueueEntry Elements with a @ <i>SubmissionTime</i> older than or equal to this dateTime are provided in the Queue Element. If not specified, there is no dateTime lower bound on candidates.
<i>NewerThan</i> ?	dateTime	Only QueueEntry Elements with a @ <i>SubmissionTime</i> newer than or equal to this dateTime are provided in the Queue Element. If not specified, there is no dateTime upper bound on candidates.
<i>QueueEntryDetails</i> ?	enumeration	Refines the level of provided information about the Queue. Allowed values are: <i>None</i> – Do not fill in the QueueEntry Elements into the Queue. <i>Brief</i> – Provide all available QueueEntry information except for the associated JobPhase Element. <i>JobPhase</i> – Provide all available QueueEntry information including the associated JobPhase Elements.
<i>QueueEntryIDs</i>	NMTOKENS	Defines an explicit list of queue entries. If not specified, all entries in the Queue are considered.
<i>StatusList</i> ?	enumerations	Only QueueEntry Elements with a @ <i>Status</i> matching one of the entries in @ <i>StatusList</i> SHALL be returned. If not specified, there is no filtering on QueueEntry/@ <i>Status</i> . Allowed values are from: QueueEntry/@ <i>Status</i> (Table 5-96, “QueueEntry Element” on page 277).
Part *	element	Return only QueueEntry Elements with all specified Part Elements. If not specified, there is no filtering on QueueEntry/Part.

5.12 Gang Jobs

XJMF provides a mechanism to specify groups of QueueEntry Elements within a queue that are processed together in a gang. A Job is submitted to a gang by specifying QueueSubmissionParams/@*GangPolicy*. The details of how individual job parts are ganged are device specific. For a description of planned job ganging, see also Section 6.4.24, “SheetOptimizing” on page 385.

Table 5-98: Messages for Gang Jobs

Messages	Description
CommandForceGang	A gang is forced to execute.
ResponseForceGang	
CommandGangStatus	The status of a gang is queried.
ResponseGangStatus	

5.12.1 ForceGang

The ForceGang Message forces all QueueEntry [*@Status* = "Waiting"] Elements that belong to a gang (as specified below) to be executed, even though the Device dependent queue entry collecting algorithm might not be completed. A QueueEntry belongs to a gang if QueueEntry/@GangName is included in the list of GangCmdFilter/@GangNames.

5.12.1.1 CommandForceGang

Table 5-99: CommandForceGang Message

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@AgentName.
<i>AgentVersion</i> ?	string	See Message/@AgentVersion.
<i>Author</i> ?	string	See Message/@Author.
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID.
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ICSVersions.
<i>ID</i>	ID	See Command/@ID.
<i>PersonalID</i> ?	NMTOKEN	See Message/@PersonalID.
<i>RelatedCommands</i> ?	NMTOKENS	See Command/@RelatedCommands.
<i>Time</i> ?	dateTime	See Message/@Time.
<i>TransactionID</i> ?	NMTOKEN	See Command/@TransactionID.
<i>GangCmdFilter</i>	element	Defines the gang(s) to be force executed.

5.12.1.1.1 Element: GangCmdFilter

Table 5-100: GangCmdFilter Element

Object Type	Element Name	Description
<i>GangNames</i> ?	NMTOKENS	@GangName of the gang(s) being queried.

5.12.1.2 ResponseForceGang

Table 5-101: ResponseForceGang Message (Sheet 1 of 2)

Name	Data Type	Description
<i>AgentName</i> ?	string	See Message/@AgentName.
<i>AgentVersion</i> ?	string	See Message/@AgentVersion.
<i>Author</i> ?	string	See Message/@Author.
<i>DeviceID</i> ?	NMTOKEN	See Message/@DeviceID.
<i>ICSVersions</i> ?	NMTOKENS	See Message/@ICSVersions.
<i>ID</i> ?	ID	See Response/@ID.
<i>PersonalID</i> ?	NMTOKEN	See Message/@PersonalID.
<i>refID</i> ?	NMTOKEN	See Response/@refID.
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode.
<i>Subscribed</i> ?	boolean	See Response/@Subscribed.

Table 5-101: ResponseForceGang Message (Sheet 2 of 2)

Name	Data Type	Description
Time ?	dateTime	See Message/@Time.
Notification ?	element	See Response/Notification.

5.12.2 GangStatus

GangStatus returns a description of the gang(s). Details are specified in GangInfo Element.

5.12.2.1 CommandGangStatus

Table 5-102: CommandGangStatus Message

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID	ID	See Command/@ID.
PersonalID ?	NMTOKEN	See Message/@PersonalID.
RelatedCommands ?	NMTOKENS	See Command/@RelatedCommands.
Time ?	dateTime	See Message/@Time.
TransactionID ?	NMTOKENv	See Command/@TransactionID.
GangQuFilter ?	element	Defines a filter for the gang(s) that are queried. If GangQuFilter is not supplied, all gangs are queried.

5.12.2.1.1 Element: GangQuFilter

Table 5-103: GangQuFilter Element

Name	Data Type	Description
GangNames ?	NMTOKENS	@GangName of the gang(s) being queried.

5.12.2.2 ResponseGangStatus

Table 5-104: ResponseGangStatus Message (Sheet 1 of 2)

Name	Data Type	Description
AgentName ?	string	See Message/@AgentName.
AgentVersion ?	string	See Message/@AgentVersion.
Author ?	string	See Message/@Author.
DeviceID ?	NMTOKEN	See Message/@DeviceID.
ICSVersions ?	NMTOKENS	See Message/@ICSVersions.
ID ?	ID	See Response/@ID.

Table 5-104: ResponseGangStatus Message (Sheet 2 of 2)

Name	Data Type	Description
PersonalID ?	NMTOKEN	See Message/@PersonalID.
refID ?	NMTOKEN	See Response/@refID.
ReturnCode ?	integer	See Response/@ReturnCode.
Subscribed ?	boolean	See Response/@Subscribed.
Time ?	dateTime	See Message/@Time.
GangInfo *	element	Describes the status of the gang(s).
Notification ?	element	See Response/Notification.

5.12.2.2.1 Element: GangInfo

Details of the gang are specified in GangInfo Elements. GangInfo is a placeholder for future gang related information.

Table 5-105: GangInfo Element

Name	Data Type	Description
GangName	NMTOKEN	Name of the gang.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="Circus"
     TimeStamp="2005-07-25T12:32:48+02:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:Circus="Circus Schema URI">
  <Query ID="Q1" Type="Circus:IsClownHappy" xsi:type="QueryIsClownHappy">
    <Circus:ClownParams Gender="male"/>
  </Query>
</JMF>

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="Circus 2"
     TimeStamp="2005-07-25T12:32:48+02:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:Circus="Circus Schema URI">
  <Response ID="M1" Type="Circus:IsClownHappy" xsi:type="ResponseIsClownHappy"
            refID="Q1">
    <Circus:Clown happy="true" name="Joe"/>
    <Circus:Clown happy="false" name="John"/>
  </Response>
</JMF>

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="IFRA"
     TimeStamp="2003-07-25T12:32:48+02:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:IFRA="IfraTrack URI">
  <Query ID="Q1" Type="IFRA:IMF" xsi:type="QueryIMF">
    <imf:IMF xmlns:imf="IfraTrack URI">
      Whatever you want (might be multiple top level Elements)
    </imf:IMF>
  </Query>
</JMF>
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="IFRA"

```

```
TimeStamp="2003-07-25T12:32:48+02:00"
MaxVersion="1.4" Version="1.4"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:IFRA="IfraTrack URI">
<Response ID="M1" Type="IFRA:IMF" xsi:type="ResponseIMF" refID="Q1">
    <imf:IMF xmlns:imf="IfraTrack URI">
        The appropriate IFRA response(s)
    </imf:IMF>
</Response>
</JMF>
```

Chapter 6 Processes

6.1 The following chapter describes the Processes that are defined in detail for XJDF. Combining Individual Process Steps

The Processes described in Chapter 6, “Processes” on page 355 define individual workflow steps that are assumed to be executed by a single-purpose Device. Many Devices, however, are able to combine the functionality of multiple single-purpose Devices and execute more than one Process. For example, a digital printer might be able to execute the **Interpreting**, **Rendering** and **DigitalPrinting** Processes. To accommodate such Devices, XJDF allows Processes to be grouped within one XJDF Node. Each XJDF Node SHALL contain a **@Types** Attribute, which in turn contains an ordered list of values of each of Processes that the Node specifies. The ordering of the Process names in the **@Types** Attribute specifies the ordering in which the Processes SHOULD be executed. If the final product result would be indistinguishable, the Device MAY change the execution order of the Processes from that given in the **@Types** Attribute.

6.1.1 Nodes with Multiple Processes of the Same Type

A Node MAY contain multiple instances of the same Process type (e.g., **@Types** = "Cutting Folding Cutting"). In this case, the ordering and mapping of Assets to Processes is significant — the parameters of the first **Cutting** Process are most likely to be different from those of the second **Cutting** Process. If multiple Processes that consume identical Resources are specified in **@Types**, the ordering of the respective ResourceSet Elements maps to the ordering in the **@Types** list, e.g. the first ResourceSet[@Name="Cutting"] applies to the first Cutting process in the list and the second ResourceSet[@Name="Cutting"] applies to the second Cutting process in the list.

6.2 Process Template

Processes are defined by their input and Output Resources, therefore, the requirements for the individual processes are provided in the tables below. Table 6-2 provides a list of resources that are valid for any process.

Note: each entry provides the requirements in the format

- Name: The ResourceSet/@**Name** of the resource, e.g: **Media** or **RunList**;
- (**ProcessUsage**): If present, the ResourceSet/@**ProcessUsage** of the resource, e.g Document or Marks in case of a **RunList**.

* Cardinality. The cardinality of the resource *, ?, + or empty. See Table 1-3, “Cardinality Symbols”: for details.

Table 6-1: Generic Input Resources (Sheet 1 of 2)

Name	Description
ApprovalDetails *	Any number of ApprovalDetails Resources MAY be appended to Processes in order to model proofing and verification requirements. This is implied and not specified explicitly in the tables in the following section. For more information on the Approval Process, see Section 6.3.1, “Approval”.
Color ?	Color Identifies all colors that are used in the Job. Color may include separations that represent die lines or other auxiliary colors.
Contact *	Employee that is associated with processing this Node.
CustomerInfo ?	CustomerInfo specifies information about the direct customer.
CustomerInfo (EndCustomer) ?	CustomerInfo (EndCustomer) specifies information about the end customers in a subcontracting situation where the direct customer is not the end customer.
Device *	Device that is associated with processing this Node.
MiscConsumable *	Generic consumable resources that are associated with processing this Node.

Table 6-1: Generic Input Resources (Sheet 2 of 2)

Name	Description
NodeInfo ?	NodeInfo specifies scheduling information about the explicit process described by this XJDF.
NodeInfo (EndCustomer) ?	NodeInfo specifies scheduling information from the end customers in a subcontracting situation where the direct customer is not the end customer.
PreflightReport *	Any number of PreflightReport Resources MAY be appended to Processes in order to convey the results of previous preflighting steps. This is implied and not specified explicitly in the tables in the following section. For more information on the Preflight Process, see Section 6.4.27, “Preflight”.
Preview *	Any number of previews MAY be associated with a Process and used for display purposes. Preview /@ PreviewUsage SHOULD be “Thumbnail” or “Viewable”.
Tool *	Miscellaneous reusable tool required for a Process.
TransferCurve ?	TransferCurve specifies area coverage correction and coordinate transformations.
UsageCounter *	Devices MAY use counters, called “usage counters”, to track equipment utilization or work performed, such as impressions produced or documents generated.

Table 6-2:

Name	Description
<i>Resource-Name</i> <i>Cardinality</i>	<i>Information about the Output Resource.</i>
<i>Resource-Name</i> (<i>someValue</i>) <i>Cardinality</i>	<i>Information about the Output Resource</i> Note: @ ProcessUsage Attribute of the specified Resource SHALL match the “ <i>someValue</i> ” value specified in the parentheses. When a Process potentially contains multiple Output Resources of the same type, the value of @ ProcessUsage distinguishes the Resources.

6.3 General Processes

6.3.1 Approval

The **Approval** Process can take place at various steps in a workflow. For example, a Resource (e.g., a printed Sheet or a finished book) is used as the input to be approved, and an **ApprovalDetails** (given, for example, by a customer or foreman) is produced. Combining the **Approval** Process with any other Process can be used to represent a request for a receipt. The Process that follows the **Approval** Process in the workflow chain will most often require the **ApprovalDetails** as Input.

Table 6-3: Approval – Input Resources

Name	Description
ApprovalParams ?	Details of the approval Process.
Resource *	The Resources to be approved. When the Input Resource of an Approval Process is a ByteMap , it is assumed that it will be displayed on a viewing Device.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-4: Approval – Output Resources

Name	Description
ApprovalDetails ?	Result of any proofing Process given, for example, by a customer or foreman.

6.3.2 Buffer

The **Buffer** Process is used to buffer a Resource for a certain time period. This can be buffering of a complete Resource or of a partial Resource (e.g., in a pipe). The *@Amount* of the input and output of Resources SHALL be equal. Waiting for printed material to dry before finishing is an example of the **Buffer** Process.

Table 6-5: Buffer – Input Resources

Name	Description
BufferParams ?	The parameters (e.g., times and locations of the Buffer Process).
Resource	The Resource Element to be buffered. These MAY be any Resource Element.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-6: Buffer – Output Resources

Name	Description
Resource	The same Resource after buffering. The Resource SHALL be a Resource.

6.3.3 Delivery

This Process can be used to describe the delivery of an end Product, a Resource to or from a location. Delivery of data over the network is specified in the **Delivery** Process.

If only the Delivery Address of the final Customer is required and no specific details of the Delivery are known, then the Delivery process need not be specifically parametrized and the Delivery address SHOULD be specified in **Contact**[@ContactTypes includes("Delivery")]/Address.

Delivery to multiple destinations or in multiple steps SHOULD be specified in **DeliveryParams** that are partitioned by *@DropID*. Common delivery of multiple products to the same address SHALL be specified by providing multiple **DeliveryParams**/DropItem elements.

Table 6-7: Delivery – Input Resources

Name	Description
DeliveryParams ?	Necessary information about the physical item or items to be delivered is stored here.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-8: Delivery – Output Resources

Name	Description
ResourceSet +	Any Resources delivered from a location. These SHALL be Resource Elements.

6.3.4 ManualLabor

This Process can be used to describe any Process where Assets are handled manually. The **ManualLabor** Process is designed to monitor any type of non-automated labor from an MIS system.

Table 6-9: ManualLabor – Input Resources

Name	Description
ManualLaborParams ?	Details on the ManualLabor Process.
ResourceSet *	Resources that are used to create the Output Resource.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-10: ManualLabor – Output Resources

Name	Description
ResourceSet *	The Resources that were created by manual work. In general these will be Component Resources, but Resource Elements MAY also be processed manually. If no Output elements are specified, the ManualLabor Process describes incidental work.

6.3.5 QualityControl

This Process defines the setup and frequency of quality controls for a Process. **QualityControl** is generally performed on **Component** Resources produced as intermediate or final output of a Process.

Table 6-11: QualityControl – Input Resources

Name	Description
QualityControlParams	Detailed definition of the QualityControl Process.
Resource	The Resource to be quality controlled. In general this will be a Component Resource.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-12: QualityControl – Output Resources

Name	Description
QualityControlResult ?	Results of the Process (e.g., measurement statistics).
Resource	This Resource describes the Resource after QualityControl has been applied.

6.3.6 Verification

The **Verification** Process is used to confirm that a Process has been completely executed. In the case of variable data printing in which every document is unique and validated individually, database access is REQUIRED. Verification in this situation can involve scanning the physical Sheet and interpreting a bar code or alphanumeric characters. The decoded data can then be either recorded in a database to be later cross referenced with a verification list, or cross referenced and validated immediately in real time. The actual database connection is implementation specific.

Verification differs from **QualityControl** in that **Verification** verifies the existence of a given set of Resources, whereas **QualityControl** verifies that the existing Resources fulfill certain quality criteria.

Table 6-13: Verification – Input Resources

Name	Description
FileSpec (Verification) ?	Reference to a file that contains implementation specific descriptions of the Resources to be verified.
Resource ?	The Resource to be verified. The input will most often be a Resource (e.g., Component).
VerificationParams ?	Controls the verification requirements.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-14: Verification – Output Resources

Name	Description
ApprovalDetails ?	Signature file that defines verification success.
FileSpec (Accepted) ?	Reference to a file that contains implementation specific descriptions of the Resources that were correctly verified.
FileSpec (Rejected) ?	Reference to a file that contains implementation specific descriptions of the Resources that were NOT correctly verified.
FileSpec (Unknown) ?	Reference to a file that contains implementation specific descriptions of the Resources that were scanned but NOT in the explicit or implied list of known Resources.
Resource	The Resource after verification. Most often the Resource will not be modified by Verification . It has been added here to allow modeling of Verification in a Combined Processes.

6.4 Prepress Processes

6.4.1 AssetListCreation

The purpose of this Process is to provide a listing of all assets and their dependent assets that are REQUIRED in order to use the input assets. This Process analyzes the input **RunList** to find dependent assets to provides a complete listing of files in the output **RunList**. **AssetListCreation** does not package, encode or compress the list of files.

Table 6-15: AssetListCreation – Input Resources

Name	Description
AssetListCreationParams ?	Parameters of the AssetListCreation Process
RunList ?	List of assets used to create a listing of dependent assets.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-16: AssetListCreation – Output Resources

Name	Description
RunList ?	A listing of all assets that the assets listed in the input RunList are dependent on including the input assets. The dependent assets are to be inserted into the output RunList as RunList/@DependencieRefs . @DependencieRefs refers to RunList elements that represent the dependent assets.

6.4.2 Bending

The **Bending** Device consumes a printing plate and bends and/or punches it. In contrast to commercial printing, for newspaper printing this Process is not integrated into the **ImageSetting** Process. An inline plate puncher SHOULD be modelled as a Combined Process consisting of **ImageSetting** and **Bending** Processes.

Table 6-17: Bending – Input Resources

Name	Description
BendingParams ?	List of assets used to create a listing of dependent assets.
ExposedMedia ?	The ExposedMedia Resource to be bent/punched. Dummy Forms are also described as ExposedMedia even though they NEED NOT be imaged.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-18: Bending – Output Resources

Name	Description
ExposedMedia ?	The bent/punched ExposedMedia Resource.

6.4.3 ColorCorrection

ColorCorrection is the Process of modifying the specification of colors in documents to achieve some desired visual result. The Process might be performed to ensure consistent colors across multiple files of a Job or to achieve a specific design intent (e.g., “brighten the image up a little”).

ColorCorrection is distinct from **ColorSpaceConversion**, which is the process of changing how the colors specified in the Job will be produced on paper. Rather, **ColorCorrection** is the process of modifying the desired result, whatever the specified color space might be.

The **ColorCorrection** Process MAY be part of a Combined Process with the **ColorSpaceConversion** Process, in which case the source and destination profiles used by the **ColorSpaceConversion** Process would be supplied from **ColorSpaceConversionParams**. Either the direct **@Adjustment** Attribute or the ICC profile Attribute **ColorCorrectionOp/FileSpec** with **@ResourceUsage = "AbstractProfile"** can be used in this scenario to apply color corrections in the Device independent ICC Profile Connection Space interpreted from the ICC source profile before the ICC destination profile is applied.

Alternatively, a **ColorCorrection** Process MAY occur after a **ColorSpaceConversion** Process. In this scenario only the **ColorCorrectionOp/FileSpec** with **@ResourceUsage = "DeviceLinkProfile"** supplied in **ColorCorrectionOp** is used.

Table 6-19: ColorCorrection – Input Resources

Name	Description
ColorantControl ?	Identifies the assumed color model for the Job.
ColorCorrectionParams ?	Parameters of the ColorCorrection Process
RunList ?	List of content elements that are to be operated on.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-20: ColorCorrection – Output Resources

Name	Description
RunList ?	List of color-corrected content elements.

6.4.4 ColorSpaceConversion

ColorSpaceConversion, as the name implies, is the process of converting all colors used in the Job to a known color space. There are two ways in which a Controller can use this Process to accomplish the color conversion. It can simply order the colors to be converted by the Device assigned to the task, or it can request that the Process simply tag the input data for eventual conversion. Additionally, the Process can remove all tags from the content.

The parameters of this Resource provide the ability to selectively control the conversion or tagging of raster data or graphical objects based on object class and/or incoming color space.

Like all other color manipulation supported in **XJDF**, the color conversion controls are based on the use of ICC profiles. While the assumed characterization of input data can take many forms, each can internally be represented as an ICC profile. In order to perform the transformations, input profiles SHALL be paired with the identified final target Device profile to create the transformation.

In order to avoid the loss of black color fidelity resulting from the transformation from a four-component CMYK to a three-component interchange space, the Controller MAY select a **DeviceLink**¹ transform as the transform to be applied when converting from a specific source colorspace to the final target device colorspace specified for the **ColorSpaceConversion** operation being applied. In these instances, the final target profile is ignored.

Table 6-21: ColorSpaceConversion – Input Resources

Name	Description
ColorantControl ?	Identifies the assumed color model for the Job. The ColorantControl Resource MAY be modified by a ColorSpaceConversion Process.
ColorSpaceConversionParams ?	Parameters that define how color spaces will be converted in the file.
RunList ?	List of pages, Sheets or byte maps on which to perform the selected operation.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-22: ColorSpaceConversion – Output Resources

Name	Description
RunList ?	List of pages, Sheets or byte maps on which the selected operation has been performed.

6.4.5 ContoneCalibration

This Process specifies the process of contone calibration. It consumes contone raster data such as that output from an **Interpreting** and **Rendering** Process. It produces contone raster data which has been calibrated to a press using a well defined screening Process.

Table 6-23: ContoneCalibration – Input Resources (Sheet 1 of 2)

Name	Description
RunList ?	Ordered list of rasterized byte maps representing pages or surfaces.
ScreeningParams ?	Parameters specifying which halftoning mechanism is to be applied and with what specific controls.

1. A **DeviceLink** transform is a transform that is defined in an ICC profile file [ICC.1] that maps directly from one specific source color space to a specific destination device color space. An example of this is a transform that maps directly from PDL source objects defined using sRGB directly to SWOP CMYK

Table 6-23: ContoneCalibration – Input Resources (Sheet 2 of 2)

Name	Description
TransferFunctionControl ?	Specifies which calibration to apply.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-24: ContoneCalibration – Output Resources

Name	Description
RunList ?	Ordered list of rasterized byte maps representing calibrated pages or surfaces.

6.4.6

6.4.7

6.4.8 DieDesign

This Process describes the design of a die tool set with one or more stations starting from a **DieLayout**.

Table 6-25: DieDesign – Input Resources

Name	Description
DieLayout ?	A Resource describing the die cutter layout. This layout is already imposed for a specific sheet size and MAY describe multiple stations.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-26: DieDesign – Output Resources

Name	Description
DieLayout *	A set of Resources describing the die cutter tool set.

6.4.9 DieLayoutProduction

This process describes the layout of one or more structural designs for a given **Media**. The output of this process is a **DieLayout** Resource, describing a tool set for the die cutter machine. The **DieLayoutProduction** Process can be performed by a human operator using a CAD application. In some cases it can be an automated Process. The Process can be run in estimation mode in which case multiple solutions are returned that can then be used as input of a cost estimation module to determine the optimal layout. The **DieDesign** Process is the packaging equivalent of a **Stripping** Process in conventional Printing. The output **DieLayout** of **DieLayoutProduction** is typically the input of a subsequent **DieDesign** Process.

Table 6-27: DieLayoutProduction – Input Resources

Name	Description
DieLayoutProductionParams ?	The parameters for the DieLayoutProduction .
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-28: DieLayoutProduction – Output Resources

Name	Description
DieLayout *	A Resource describing the die cutter tool set. When the process is run in estimation mode, multiple alternative DieLayout Elements are returned, otherwise a single DieLayout is generated.

Example 6-1: DieLayoutProduction: Single Shape and Two Sheet SizesExample of **DieLayoutProduction** of a single shape on 2 stock sheet sizes

TBD 2.x Example.

```
<!-- DieLayoutProduction Sample
    Date:Sept 2007 Version: 1.00
    Single Shape is repeated on a range of alternative sheet sizes.
-->
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n001"
    Type="DieLayoutProduction" Status="Waiting" JobPartID="ID234"
    DescriptiveName="Single shape versus a set of sheet sizes"
    Version="1.4">
<ResourcePool>
    <ShapeDef Class="Parameter" ID="Shape1Up" Status="Available">
        <FileSpec URL="file://myserver/myshare/olive.dd3"/>
    </ShapeDef>
    <!-- Layout can chose from 2 stock sheet sizes. Nesting with 2nd row
        rotated and secondary gutters. Rotate against grain/flute
        is not allowed.
    -->
    <DieLayoutProductionParams Class="Parameter" ID="LayParam"
        Status="Available">
        <ConvertingConfig SheetWidth="2834.64 ~ 2834.64"
            SheetHeight="2267.72 ~ 2267.72"/>
        <ConvertingConfig SheetWidth="3401.57 ~ 3401.57"
            SheetHeight="2834.64 ~ 2834.64"/>
        <RepeatDesc GutterY="0.0" GutterY2="14.17" AllowedRotate="None"
            LayoutStyle="Reverse2ndRow"/>
    </DieLayoutProductionParams>
    <!-- The layout with minimum waste will be returned as the final result. -->
    <DieLayout Class="Parameter" ID="DieLay" Status="Unavailable"/>
</ResourcePool>
<ResourceLinkPool>
    <ShapeDefLink rRef="Shape1Up" Usage="Input"/>
    <DieLayoutProductionParamsLink rRef="LayParam" Usage="Input"/>
    <DieLayoutLink rRef="DieLay" Usage="Output"/>
</ResourceLinkPool>
</JDF>
```

Example 6-2: DieLayoutProduction: Single Shape and Range of Sheet SizesExample of **DieLayoutProduction** of a single shape on a range of sheet sizes. The sheet sizes have defined minimum and maximum width and height. The layout is optimized for a particular order quantity

TBD 2.x Example.

```
<!-- DieLayoutProduction Sample
    Date:Sept 2007 Version: 1.00
    Single Shape is repeated on a continuous range of sheet sizes. -->
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n001"
    Type="DieLayoutProduction" Status="Waiting"
```

```

DescriptiveName="Single shape versus a set of sheet sizes"
JobPartID="ID400" Version="1.4">
<ResourcePool>
  <ShapeDef Class="Parameter" ID="Shape1Up" Status"Available">
    <FileSpec URL="file://myserver/myshare/olive.dd3"/>
  </ShapeDef>
  <!-- Layout can choose sheet sizes between 1200mm-1000mm wide and
      1000mm-800mm high. The layout will be optimized for order quantities
      of 1 million boxes. Gutters are 5mm and cross flute/grain rotation
      is not allowed.
  -->
  <DieLayoutProductionParams Class="Parameter" ID="LayParam"
    Status"Available">
    <ConvertingConfig SheetWidth="3401.57 ~ 2834.64"
      SheetHeight="2834.64 ~ 2267.72"/>
    <RepeatDesc OrderQuantity="1000000" GutterX="14.17" GutterY="14.17"
      AllowedRotate"None"/>
  </DieLayoutProductionParams>
  <!-- The layout with minimum waste will be returned as the
      final result. -->
  <DieLayout Class="Parameter" ID="DieLay" Status"Unavailable"/>
</ResourcePool>
<ResourceLinkPool>
  <ShapeDefLink rRef="Shape1Up" Usage"Input"/>
  <DieLayoutProductionParamsLink rRef="LayParam" Usage"Input"/>
  <DieLayoutLink rRef="DieLay" Usage"Output"/>
</ResourceLinkPool>
</JDF>

```

Example 6-3: DieLayoutProduction: Two Shapes and Range of Sheet Sizes

Example of **DieLayoutProduction** of 2 shapes on a range of sheet sizes. The sheet sizes have defined minimum and maximum width and height. The layout is optimized for a particular order quantity of 2 boxes.

TBD 2.x Example.

```

<!-- DieLayoutProduction Sample
     Date:Sept 2007 Version: 1.00
     2 Shapes is repeated on a continuous range of sheet sizes.
-->
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n001"
  Type="DieLayoutProduction"
  Status"Waiting"
  DescriptiveName="Single shape versus a set of sheet sizes"
  Version="1.4">
<ResourcePool>
  <ShapeDef Class="Parameter" ID="Shape1Up" Status"Available">
    <FileSpec URL="file://myserver/myshare/beef.dd3"/>
  </ShapeDef>
  <ShapeDef Class="Parameter" ID="Shape1Up2" Status"Available">
    <FileSpec URL="file://myserver/myshare/chicken.dd3"/>
  </ShapeDef>
  <!-- Layout can chose sheetsizes between 1200mm-1000mm wide and
      1000mm-800mm high. Layout is optimized for an order
      quantity of 300k boxes for beef and 700k boxes for chicken.
      Gutters are 5mm and cross flute/grain rotation is not allowed.
  -->
  <DieLayoutProductionParams Class="Parameter" ID="LayParam"
    Status"Available">

```

```

<ConvertingConfig SheetWidth="3401.57 ~ 2834.64"
    SheetHeight="2834.64 ~ 2267.72"/>
<RepeatDesc OrderQuantity="300000" GutterX="14.17" GutterY="14.17"
    AllowedRotate="None"/>
<RepeatDesc OrderQuantity="700000" GutterX="14.17" GutterY="14.17"
    AllowedRotate="None"/>
</DieLayoutProductionParams>
<!-- The layout with minimum waste will be returned as the final
    result. -->
<DieLayout Class="Parameter" ID="DieLay" Status="Unavailable"/>
</ResourcePool>
<ResourceLinkPool>
    <ShapeDefLink rRef="Shape1Up" Usage="Input"/>
    <ShapeDefLink rRef="Shape1Up2" Usage="Input"/>
    <DieLayoutProductionParamsLink rRef="LayParam" Usage="Input"/>
    <DieLayoutLink rRef="DieLay" Usage="Output"/>
</ResourceLinkPool>
</JDF>

```

6.4.10 ImageEnhancement

The ***ImageEnhancement*** Process describes generic image data processing.

Note: the source MAY be any image, but also text or vector graphics.

Table 6-29: ImageEnhancement – Input Resources

Name	Description
ImageEnhancementParams ?	Describes the controls selected for the manipulation of images.
RunList ?	List of content data elements on which to perform the selected operations.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-30: ImageEnhancement – Output Resources

Name	Description
RunList ?	List of page contents with images that have been manipulated as indicated by the ImageEnhancementParams Resource.

6.4.11 ImageSetting

The ***ImageSetting*** Process is executed by an imagesetter or platesetter that images a bitmap onto the film or plate media.

Table 6-31: ImageSetting – Input Resources (Sheet 1 of 2)

Name	Description
ColorantControl ?	The ColorantControl Resources that define the ordering and usage of inks during marking on the imagesetter.
DevelopingParams ?	Controls the physical and chemical specifics of the media development process.

Table 6-31: ImageSetting – Input Resources (Sheet 2 of 2)

Name	Description
ExposedMedia ?	When imaging to reusable media, ExposedMedia MAY also be used as input to ImageSetting . Constraint: exactly one of Media or ExposedMedia SHALL be specified.
ImageSetterParams ?	Controls the Device specific features of the imagesetter.
Media ?	The unexposed media. Constraint: exactly one of Media or ExposedMedia SHALL be specified.
RunList ?	Identifies the set of bitmaps to image. MAY contain bytemaps or images.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-32: ImageSetting – Output Resources

Name	Description
ExposedMedia	The exposed media Resource. In case of plate setting, this is the exposed set of plates. In case of film setting, this is the exposed set of films.

6.4.12 Imposition

The **Imposition** Process is responsible for combining pages of input graphical content onto surfaces whose dimensions are reflective of the physical output media. Static or dynamic printer's marks can be added to the surface in order to facilitate various aspects of the production process. Among other things, these marks are used for press alignment, color calibration, job identification and as guides for cutting and folding.

Note that the **Imposition** Process specifies the task of combining pages and marks on sheets. The task of setting up the parameters needed for **Imposition** (e.g., creating the **Layout** Resource) is defined by **Stripping**.

Table 6-33: Imposition – Input Resources

Name	Description
Layout ?	A Layout Resource that indicates how the content pages from the Document RunList and marks from the Marks RunList (see below) are combined onto imposed surfaces.
RunList (Document) ?	Structured list of incoming page contents which is transformed to produce the imposed surface images.
RunList (Marks) ?	Structured list of incoming marks. These are typically printer's marks such as fold marks, cut marks, punch marks or color bars.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-34: Imposition – Output Resources

Name	Description
RunList ?	Structured list of imposed surfaces. The @ElementType of the RunList Resource SHALL be " <i>Surface</i> ". Typically the output RunList will be Partitioned by @PartIDKeys = " <i>SheetName Side Separation</i> ". If the Imposition Process is executed before RIPing , this will generally be consumed by an Interpreting Process. In the case of post-RIP Imposition , it will be consumed by DigitalPrinting or ImageSetting .

There are two mechanisms provided for controlling the flow of page images onto sheet surfaces:

The default mechanism is for non-automated (e.g., fully-specified) **Imposition**, which originally derived from **Layout** in PJTF. Fully-specified imposition explicitly identifies all page content for each sheet imaged and references these pages by means of the order in which they are defined in the input **RunList (Document)** Resource. Static printer's marks are referenced in a similar fashion from the input **RunList (Marks)** Resource.

Setting the **@Automated** attribute of the **Layout** Resource to "true" activates a template approach to imposition and relies upon the full hierarchy structure of the document (as specified by the **RunList (Document)** and referenced **Structured PDL** data) to specify the page content to be imposed.

In **XJDF**, there is a single **Layout** Resource definition. Its structure is broad enough to encompass the needs of both fully specified and template-driven imposition. When described fully (**@Automated = "false"**), the **Layout** Resource Partition structure defines the imposition to take place. The highest level of each Partition defines a signature. The children of each of the signatures in turn specifies an array of sheets, and each sheet MAY have up to two surfaces (Front and/or Back), on which the page images and any printer's marks are to be placed using **PlacedObject** Elements. A sheet that specifies no surface content SHALL be interpreted as blank. Pages that are to be printed SHALL be placed onto surfaces using **ContentObject** Subelements which explicitly identify the page (Typically done using the **ContentObject/@Ord** Attribute which specifies an index into the document **RunList**). Thus, the **Layout** Partition hierarchy SHALL explicitly specify which pages are to be imaged onto each surface.

For JDF 1.3, automated imposition was originally defined such that **Layout** Resource Partitions specified a single signature of sheet(s) upon which page content was to be imposed. The sequence of pages to be imaged via automated imposition was defined by the Document **RunList**. The pages were pulled from this sequence as needed in order to satisfy the **ContentObject** Elements defined for each sheet surface in the signature of the **Layout** Resource. The signature was repeated as necessary until all pages available in the Document **RunList** had been used.

Note that the XML order in which the Partitions of the **Layout** Resource are defined is significant for both automated and non-automated imposition and defines the order in which the imposition engine processes the **RunList (Document)**.

6.4.12.1 Glossary for Automated Imposition

This table below introduces terms and concepts necessary for understanding automated imposition processing.

Table 6-35: Glossary for Automated Imposition (Sheet 1 of 4)

Term	Definition
Base Index	When processing an Imposition Template , the imposition engine maintains an internal Base Index into the Page Pool being processed. That Base Index is added to the ContentObject/@Ord value, resulting in an index into the Page Pool for referencing the page to be placed, and is updated for each Imposition Template iteration. Both positive and negative base indices are maintained for use when ContentObject/@Ord has either a negative or positive value. For an example, see Example O-31, "Algorithm for Processing an Imposition Template" on page 1330.
Base Ord	Same as Base Index .
Collect	Set of sheets that are collected together prior to gathering.
Document Major Processing Order	<p>Document Major Processing Order refers to the scenario wherein all instances of a given document class (across all sets to be processed) SHALL be produced before starting processing for the next document class.</p> <p>For instance, the production requirements may state that all brochures SHALL be produced for each set, followed by all cover letters and then all postcards. This processing order is an example of Document Major.</p>

Table 6-35: Glossary for Automated Imposition (Sheet 2 of 4)

Term	Definition
Imposed Sheet Set	Describes a single set of sheet definitions generated by the imposition engine containing imposed content. Note that this may represent a precut set of sheets in a cut-and-stack workflow (where the maximum number of sheets in the Imposed Sheet Set is defined by Layout/LogicalStackParams/@MaxStackDepth), or a collect when no Logical Stacks are defined.
Imposition Template	A first-level branch of a Partitioned Layout Resource having @Automated = "true" that describes a single set of sheets with a common imposition layout that accommodates very specific production characteristics. A single Layout Resource defines a collection of one or more Imposition Templates .
Instance Document	The imposition engine treats each immediate child Node of a set in a Structured PDL as an Instance Document . This is used as the basis for generating @EndOfDocument breaks in the resulting RunList (Surface) , and for processing RunList/@DocCopies Attributes (see Section 8.134, “RunList” on page 887). If a set has only pages as its children, then a single Instance Document is assumed to exist.
Logical Sheet	One or more pages placed onto a sheet definition within a Logical Stack (i.e., a sheet definition within a Logical Stack).
Logical Stack	When Layout/LogicalStackParams/@MaxStackDepth is specified in the root of the Layout Resource, then the imposition engine is configured for imposition onto multiple Logical Stacks . These stacks are described through the use of adding Layout/PlacedObject/@LogicalStackOrd to stack-specific descriptions for each placed object. For more information, see Section 6.4.18.4.1, “Using Logical Stacks” on page 382. For example usage, see Example O-34, “Booklet Using Automated Imposition” on page 1334.
Logical Stack Set	The set of Logical Stacks described by an Imposed Sheet Set .
Page Pool	<p>A Page Pool refers to a delimited sequence of pages defined within the RunList (Document) input to the Imposition Process. A Page Pool MAY encompass all pages of the RunList (Document) as in the case of Unstructured PDLs. In the case of Structured PDLs, a Page Pool is defined to be that set of pages represented by a leaf node of the document structure. For example, a brochure which has a sub-structure of Cover and Body has two leaf nodes, Cover and Body, respectively. If Body were further divided into Chapter sections, then the leaf nodes of the Brochure would be the Cover and each Body Chapter. RunList/@ElementType may be used to demote an already Structured PDL to be treated as an Unstructured PDL. Examples of Structured PDL formats include PPML, PPML/VDX, and ISO 16612-2 PDF/VT.</p> <p>Imposition Templates select Page Pools to be processed based on their Partition Keys whose values are derived from metadata present in the PDL data (e.g., Layout Partitioned by @DocTags = "Letter" would process all Page Pools of the current Set whose metadata derived Partition Key @DocTags matches "Letter"). See below for more detail.</p> <p>It is important to note that the pages in a Page Pool SHALL be presented to the imposition engine in a well defined order known to the Layout Resource creator (typically reader order) in order for them to be processed correctly.</p>

Table 6-35: Glossary for Automated Imposition (Sheet 3 of 4)

Term	Definition
Page Pool List	<p>A Page Pool List refers to a sequence of one or more Page Pools (contiguous or disjoint in the RunList (Document)) aggregated together and treated as a single Page Pool for processing by a selected Imposition Template. For example, if a Page Pool List is constructed from the Page Pools: Chapter1, Chapter2, and Chapter4 as defined in an input RunList (Document), then the aggregate result is a single pool of pages consisting of the pages from Chapter1, Chapter2 and Chapter4. The order of the pages of the Page Pool List SHALL be processed in the order in which the Page Pools are defined in the RunList (Document). The boundaries between Page Pools in a Page Pool List are implicitly maintained for use by the imposition processor for making page level sheet surface mapping decisions during processing (e.g., specifying a right side facing pages start at the beginning of each chapter). Page Pools are aggregated into Page Pool Lists through the use of the Layout/@BaseOrdReset Attribute. If @BaseOrdReset = "PagePoolList" then all Page Pools processed by the Imposition Template are aggregated. If @BaseOrdReset = "PagePool", then each Page Pool is processed separately.</p> <p>It is important to note that the pages in a Page Pool List SHALL be presented to the imposition engine in a well defined order known to the Layout Resource creator (typically reader order) in order for them to be processed correctly.</p>
PDL Metadata	<p>Various PDL formats provide for the definition of key/value pairs within the PDL that MAY be treated as metadata for the purpose of Process parameterization. For example, the metadata key/value pairs specified in the PDL data may identify the type of finished document using @DocumentType = "PostCard" or "Booklet", which would then affect the selection of which Imposition Template is to be applied.</p> <p>The Imposition Process makes use of metadata to make decisions as to which Page Pools should be processed through an Imposition Template. These decisions are performed by comparing the explicit Partition Key settings for each Imposition Template to the Partition Key/value settings mapped from the PDL for each Page Pool in the current set, and each matching Page Pool is processed through the corresponding Imposition Template(s).</p> <p>Within an Imposition Template, metadata associated with individual pages MAY also be used to parameterize dynamic mark and slug-line content generation (see example below). Refer to the RunList/MetadataMap Element definition for information on how to specify the mapping of PDL specified metadata values for use by XJDF (e.g., using Partition Keys or GeneralID keys).</p> <p>The PDL Processor SHALL make use of the RunList/MetadataMap to generate Partition Keys, GeneralID and other values during the course of imposition processing. These values SHALL be regenerated as necessary, as the metadata key/value pairs in the PDL change based on which portion of the PDL is being processed.</p>
PDL Processor	A PDL interface that hides details of a particular PDL and syntax, etc. from the imposition engine itself. Its role is to present the structure of the PDL and pools of pages within the PDL structure to the imposition engine in a PDL independent way.
Recipient Set	Set of finished pages produced for a single recipient.
Set Major Processing Order	Set Major Processing Order refers to the scenario when all documents of a set instance are produced before starting on the next set instance; this is the typical processing order for most VDP applications.
Sheet Definition	A branch of an Imposition Template that describes the imposition to be performed for a sheet. Sheet Definitions for automated imposition SHALL be partitioned by @SheetName and @Side .

Table 6-35: Glossary for Automated Imposition (Sheet 4 of 4)

Term	Definition
Structured PDL	<p>A Structured PDL defines sequences of groupings of pages. These groupings may be as simple as specifying the set of pages belonging to a chapter or cover of a booklet where such a group is a Page Pool. In the case of Variable Document Printing (VDP) Structured PDLs, there are often multiple sets of content where typically a set instance comprises the content to be delivered to a single recipient. Each set has one or more documents, and documents may be further subdivided into subdocuments in hierarchical fashion. The imposition engine processes each set individually in the sequence specified in the interpretation specified by the RunList that references the Structured PDL data file.</p> <p>The general structure of a Structured PDL is identified by the PDL (PDL specification or PDL instance) itself or the value of the RunList/@ElementType Attribute.</p>
Structured PDL – MultiDocument	<p>For MultiDocument PDL files, the PDL processor supplies the context to the imposition processor that represents the PDL's document structure. This context is defined as</p> <p>Set – represents a single set containing all of the documents in the PDL file, therefore the value of @SetIndex SHALL always be 0.</p> <p>Document – is always the first hierarchical level in the file.</p> <p>SubDoc0 to SubDoc9 – represent consecutive levels of the hierarchy below the Document level in the file not including the level representing individual pages. If any level of the hierarchy is not defined, the value of the corresponding @SubDocIndexn is undefined.</p> <p>Pages – represent individual pages in the PDL.</p>
Structured PDL – MultiSet	<p>For MultiSet PDL files, the PDL processor supplies the context to the imposition processor that represents the PDL's set and document structure. This context is defined as</p> <p>Set – represents a set of related documents.</p> <p>Document – is always the first hierarchical level below the Set level. If a MultiSet file contains only Sets with no document or sub-document breaks (no levels are defined below the Set level), all of the pages of the set are considered to be included in a single document therefore the @DocIndex is always 0.</p> <p>SubDoc0 to SubDoc9 – represent consecutive levels of the hierarchy below the Document level in the file not including the level representing individual pages. If any level of the hierarchy is not defined, the value of the corresponding @SubDocIndexn is undefined.</p> <p>Pages – represent individual pages in the PDL.</p> <p>Note: the lowest level of the XJDF hierarchy (Set, Document, SubDocn) mapped by the PDL processor represents a Page Pool context.</p>
Unstructured PDL	<p>An Unstructured PDL is a content file consisting of a single set of one or more pages. Typically such a PDL file is considered to be a single document and a single Layout Imposition Template would be applied to the entire set of pages. When an XJDF imposes structure on such a file either using direct @Page indices or a Partitioned RunList pointing to different page ranges of the file using @EndOfSet, @EndOfDocument Attributes, then the imposition engine will treat the input RunList Resource as a Structured PDL.</p>

6.4.12.2 Variables for Automated Imposition

The imposition engine maintains a set of locally scoped variables that may be referenced during imposition processing. The values of these variables reflect the current context of processing during execution of the Imposition

process. These variables include those described in Section I, “Generating strings with Format and Template” on page 1203, as well as those described in bulleted items below. All variables below are integer variables.

Table 6-36: Variables for Automated Imposition (Sheet 1 of 2)

Name	Data Type	Description
<i>CollectIndex</i> ?	integer	Represents a zero based index of the current collect of sheets being generated by an automated Imposition Template from the current Page Pool or Page Pool List being processed. May be greater than zero if Layout/@MaxCollect is specified and is greater than 1.
<i>CollectSheetIndex</i> ?	integer	Is a zero-based index of the current physical or Logical Sheet of the current collect generated by an automated Imposition Template from the current Page Pool or Page Pool List being processed. Logical Sheets are used when Logical Stacks are defined.
<i>ImposedSheetSetIndex</i> ?	integer	Is the 0-based Imposed Sheet Set index.
<i>PoolSheetIndex</i> ?	integer	Is a zero-based index of the current physical or logical sheet generated from the current Page Pool or Page Pool List within an automated Imposition Template . Logical Sheets are used when Logical Stacks are defined. The value of this variable is independent of the number of collects generated by the same automated Imposition Template .
<i>SheetCount</i> ?	integer	Is the current number of physical or logical sheets generated during the processing of the automated Layout Resource . Logical Sheets are used when Logical Stacks are defined. At the beginning of processing of the Layout Resource , the value of this variable is set to zero. The value of this variable may be reset to zero in later Layout Partitions using the Layout/@SheetCountReset Attribute. @SheetCount is always reset to zero at the beginning of processing of a set regardless of the value of Layout/@SheetCountReset .
<i>SubDocIndexn</i> ?	integer	Where <i>n</i> represents any hierarchical structure levels below the level of the current document present in the Structured PDL data to be processed. For example, @SubDocIndex0 might represent a collection of chapters in a brochure where its containing parent is at the document level (@DocIndex is used to indicate the position (index) of the document in its containing Set).
<i>TotalCollects</i> ?	integer	Is the total number of collects generated by an automated Imposition Template from the current Page Pool or Page Pool List being processed.
<i>TotalImposedSheetSets</i> ?	integer	Is the total number of Imposed Sheet Sets defined for the job.
<i>TotalSets</i> ?	integer	Is the total number of recipient sets generated for the Job. Note that in cases where it is used before the end of content imposition, it is necessary for the imposition processor to count the number of sets in the PDL content.

Table 6-36: Variables for Automated Imposition (Sheet 2 of 2)

Name	Data Type	Description
<i>TotalSheetCount</i> ?	integer	Is the total number of physical or Logical Sheets generated during the processing of the automated Layout Resource. Logical Sheets are used when Logical Stacks are defined. The value of this variable may be recalculated in later Layout Partitions using the Layout/@SheetCountReset Attribute. <i>@TotalSheetCount</i> is always reset to zero at the beginning of processing of a set regardless of the value of Layout/@SheetCountReset .
<i>TotalSheetsInCollect</i> ?	integer	Is the total number of physical or Logical Sheets that make up the current collect generated by an automated Imposition Template from the current Page Pool or Page Pool List being processed. Logical Sheets are used when Logical Stacks are defined.
<i>TotalSheetsInPool</i> ?	integer	Is the total number of physical or Logical Sheets generated from the current page pool or page pool list within an automated Imposition Template . Logical Sheets are used when Logical Stacks are defined.

The above variables MAY be used for controlling the activation of printer's marks (See **Layout/MarkObject/MarkActivation**). For example:

Example 6-4: Automated Imposition: MarkObject

This example causes a slug line to be imaged on the bottom center of the first sheet of the set of sheets comprising a signature instance. Here are the details. For **MarkActivation/@Context**, its value of "*CollectIndex*" specifies that the value of **@CollectIndex** is the index used with **MarkActivation/@Index**. For **MarkActivation/@Index**, its value of 0 specifies that the sheet receive the specified slug line if the value of **@CollectIndex** is 0 (i.e., if it is first sheet of the signature instance). **Note:** if **@Index** were "1 4 6", then the slug line would go on the second, fifth and seventh sheets.

TBD 2.x Example.

```
<MarkObject Anchor="BottomCenter" CTM="1 0 0 1 0 0">
  <DeviceMark FontSize="8" Font="MySlugLineFont"/>
  <!--Result: Gender=maile -->
  <JobField JobFormat="Gender=%s" JobTemplate="GeneralID:Gender"/>
  <RefAnchor Anchor="BottomCenter" AnchorType="Sibling" rRef="1000006"/>
  <MarkActivation Context="CollectIndex" Index="0"/>
</MarkObject>
```

6.4.12.3 Execution Model for Automated Imposition

The **Imposition** Process transforms the sequences of pages contained within a **Page Pool** or **Page Pool List** to a specific sequence of imposed sheet surfaces. The **Imposition Templates** and the order of the **Imposition Templates** defined by the **Layout** Resource explicitly define the page to sheet surface mapping transformation applied by the imposition engine.

The pseudo-code below describes the processing performed by the imposition engine at a high level:

```
For each Set in the order specified in the input RunList (Document)
  For each Imposition Template
    For each Page Pool in the Set
      If the Partition Key conditions for the Imposition Template are satisfied
        then process the Page Pool through the Imposition Template.
```

Thus, each **Layout Resource Imposition Template** is processed in the XML structure order specified. Every **Page Pool** belonging to the current set is then evaluated against the Partition Keys specified for that **Imposition Template** to determine if it is to be processed by that **Imposition Template**.

Since each **Page Pool** is evaluated for each **Imposition Template**, it is possible to reuse the same **Page Pool** with multiple **Imposition Templates**. For an example algorithm for processing **Page Pools** through an **Imposition Template**, see Example O-31, “Algorithm for Processing an Imposition Template” on page 1330.

The **RunList** Resource output from the **Imposition** Process represents a sequence of imposed sheet surfaces where each surface may be represented either by pointing to PDL content where all the input pages are imposed onto single PDL pages, or, when used with a Combined Process may refer to the page set along with imposition instructions to the interpreter using an exchange Resource. The structure of the **Layout** Resource affects the Partition Keys conserved by its output **RunList** (and its referenced content), by conserving all Partition Keys specified in the **Layout** along with generating all of the appropriate Partition Keys, such as **@SetIndex**, **@DocIndex**, **@SheetIndex**. The output **RunList** can be viewed conceptually as a collection of sheet surface pairings (front and back) that conserves information about which **Layout Imposition Template** and **Page Pool** metadata that was in scope at the time the sheets were generated.

Note: **@DocIndex** is always generated even if every set contains only a single document; a set that contains only pages is treated as a set with a single document.

Note: **MarkObject/@Ord** works in the same way for automated imposition as for non-automated imposition. In other words, the **@Ord** value corresponds to the page entry described by that absolute **@Ord** position in the **RunList** (Marks).

Example 6-5: Imposition Template: Layout

Thus, if the **Imposition Template (Layout)** in this example is applied, then the resulting **RunList** Resource conceptually conserves the following Partition Keys: **@SetIndex**, **@SheetIndex**, **@DocTags**, **@DocIndex**, **@SheetName** and **@Side** along with any other in-scope Partition Keys.

Note that in this example, **@SetIndex** and **@DocIndex** are conserved by setting **@EndOfSet** and **@EndOfDocument** respectively in the output **RunList (Surface)**. In a **Layout** that defines **Logical Stacks** containing multiple documents or sets within **Imposed Sheet Sets**, **@SetIndex** and **@DocIndex** would need to be conserved by explicitly setting the value of the **@SetIndex** and **@DocIndex** Partition Keys. The **RunList** is expected to be partitioned by **@Run**, where each **@Run** represents one or more Sheets, each having at least one surface either implied by **RunList/@SheetSides**, or explicitly Partitioned by **@Side**.

TBD 2.x Example.

```
<Layout Class="Parameter" ID="L1" Status="Available"
      PartIDKeys="DocTags SheetName Side" Automated="true">
  <Layout DocTags="CoverLetter">
    <Layout SheetName="CoverLetterSheets">
      <Layout Side="Front">
        <ContentObject Ord="0" CTM="1 0 0 1 0 0"/>
      </Layout>
    </Layout>
  <Layout DocTags="Booklet">
    <Layout SheetName="BookletSheets">
      <Layout Side="Front">
        <ContentObject Ord="0" CTM="1 0 0 1 0 0"/>
        <ContentObject Ord="-1" CTM="1 0 0 1 0 0"/>
      </Layout>
      <Layout Side="Back">
        <ContentObject Ord="1" CTM="1 0 0 1 0 0"/>
        <ContentObject Ord="-2" CTM="1 0 0 1 0 0"/>
      </Layout>
    </Layout>
  </Layout>
</Layout>
```

```
</Layout>
</Layout>
```

Example 6-6: Output RunList (Surface)

TBD 2.x Example.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n_000000"
      JobID="JobID" JobPartID="n_000000" Status="Waiting" Type="Combined"
      Types="Interpreting Rendering DigitalPrinting Stitching" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Combined">
  <AuditPool>
    <Created ID="a_000001"TimeStamp="2008-10-23T11:14:03+02:00"/>
  </AuditPool>
  <!--Generated by the CIP4 Java open source JDF Library version :
      CIP4 JDF Writer Java 1.3 BLD 52
  -->
  <ResourcePool>
    <Component Class="Quantity" ID="r_000002" Status="Unavailable"
               ComponentType="Sheet" />
    <DigitalPrintingParams Class="Parameter" ID="r_000003" Status="Available"/>
    <InterpretingParams Class="Parameter" ID="I_000001" Status="Available"/>
    <StitchingParams Class="Parameter" ID="SP_000001" Status="Available"/>
    <Media Class="Consumable" ID="r_000004" Status="Available"/>
    <RunList Class="Parameter" ID="r_000005" PartIDKeys="Run" Status="Unavailable">
      <RunList EndOfSet="true" NPage="1" Pages="0" Run="1" SheetSides="Front">
        <LayoutElement Class="Parameter" ContentDataRefs="l_000007">
          <ContentListRef rRef="r_000006"/>
        </LayoutElement>
      </RunList>
      <RunList EndOfSet="true" NPage="4" Pages="1 ~ 4" Run="2"
              SheetSides="FrontBack">
        <LayoutElement Class="Parameter" ContentDataRefs="l_000008">
          <ContentListRef rRef="r_000006"/>
        </LayoutElement>
      </RunList>
      <RunList EndOfSet="true" NPage="1" Pages="5" Run="3" SheetSides="Front">
        <LayoutElement Class="Parameter" ContentDataRefs="l_000007"/>
      </RunList>
      <RunList EndOfSet="true" NPage="4" Pages="6 ~ 9" Run="4"
              SheetSides="FrontBack">
        <LayoutElement Class="Parameter" ContentDataRefs="l_000008">
          <ContentListRef rRef="r_000006"/>
        </LayoutElement>
      </RunList>
    </RunList>
    <ContentList Class="Parameter" ID="r_000006" Status="Unavailable">
      <ContentData ID="l_000007">
        <ContentMetaData>
          <Part DocTags="CoverLetter" SheetName="CoverLetterSheet"/>
        </ContentMetaData>
      </ContentData>
      <ContentData ID="l_000008">
        <ContentMetaData>
          <Part DocTags="BrochureSheets" SheetName="BrochureSheet"/>
        </ContentMetaData>
      </ContentData>
    </ContentList>
  </ResourcePool>
</JDF>
```

```

</ResourcePool>
<ResourceLinkPool>
  <ComponentLink CombinedProcessIndex="3" Usage="Output" rRef="r_000002"/>
  <DigitalPrintingParamsLink CombinedProcessIndex="2" Usage="Input"
    rRef="r_000003"/>
  <MediaLink CombinedProcessIndex="1 2" Usage="Input" rRef="r_000004"/>
  <RunListLink CombinedProcessIndex="0 2" Usage="Input" rRef="r_000005"/>
  <InterpretingParamsLink Usage="Input" rRef="I_000001"/>
  <StitchingParamsLink Usage="Input" rRef="SP_000001"/>
</ResourceLinkPool>
</JDF>

```

6.4.12.4 Configuration for Various Automated Impositions

6.4.12.4.1 Using Logical Stacks

An **Imposed Sheet Set** output by the imposition engine can describe multiple **Logical Stacks**. Each of these **Logical Stacks** is placed onto a well-defined section of the sheet definitions, and after printing will typically be cut in a post-press finishing operation, generating the representative physical stacks.

Logical Stacks are configured through the use of two mechanisms:

- The **Layout**/**LogicalStackParams** Element specifies the control for each **Logical Stack** including how **Logical Sheets** are sequenced onto a **Logical Stack**, and restrictions on how **Logical Sheets** of **Recipient Sets** can span **Logical Stacks** and **Imposed Sheet Sets**.
- The **PlacedObject**/**@LogicalStackOrd** is used to assign individual placed object definitions to a **Logical Stack**. Each **PlacedObject** defines the CTM for placing that object onto the **Logical Stack**. Each of the **PlacedObject** Elements will have the same **@Ord** value across the **Logical Stacks**.

To define a **Logical Stack**, the **Layout**/**LogicalStackParams** Element SHALL be present in the root of the **Layout** Resource. This Element configures the imposition engine to place **Logical Sheets** within **Logical Stacks**. The maximum number of sheets that can make up an **Imposed Sheet Set** is specified by **LogicalStackParams**/**@MaxStackDepth**. Stacks are identified through the use of **LogicalStackParams**/**Stack**/**@LogicalStackOrd**; the first **Logical Stack** is **@LogicalStackOrd = "0"**, the 2nd is **"1"**, etc.

All **Logical Stacks** defined by **Layout**/**LogicalStackParams** SHALL be used in all **Imposition Templates**, with the exception of an optional sheet (see **Layout**/**SheetCondition** in Section 8.87.10, “**SheetCondition**” on page 736) having a **@Condition** of **"LogicalStackSetBegin"** or **"LogicalStackSetEnd"** – these optional **Logical Sheets** are placed into a specific **Logical Stack** as specified by the **PlacedObject**/**@LogicalStackOrd** in the optional sheet.

The imposition works by traversing each **Logical Stack** (in the sequence specified by **LogicalStackParams**/**Stack**/**@LogicalStackSequence**). Each **Imposition Template** is processed where **PlacedObject** Elements are evaluated for one of two cases:

- 1 The **PlacedObject** has no **@LogicalStackOrd**. In this case, the **PlacedObject** is considered to be a physical sheet-level object, and is placed once at the start of processing for a physical sheet. Note that only information relevant to a physical sheet (such as **@SheetIndex**) is in scope for use in generating dynamic marks. An example of a physical sheet-level mark is a cut mark for where to cut the stacks.
- 2 The **PlacedObject** has a **@LogicalStackOrd**. In this case, only **PlacedObject** Elements that have a matching **@LogicalStackOrd** for the current **Logical Stack** being processed are placed. Note that information relevant to documents and pages (such as **@CollectIndex** or **@TotalSheetsInPool**) is in scope for use in generating dynamic marks.

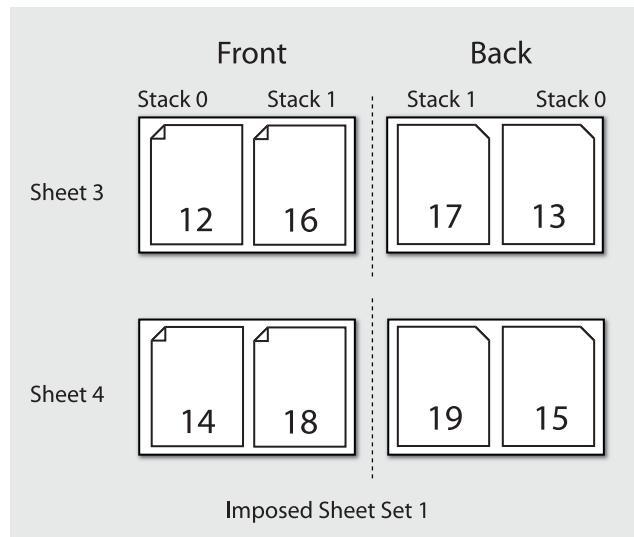
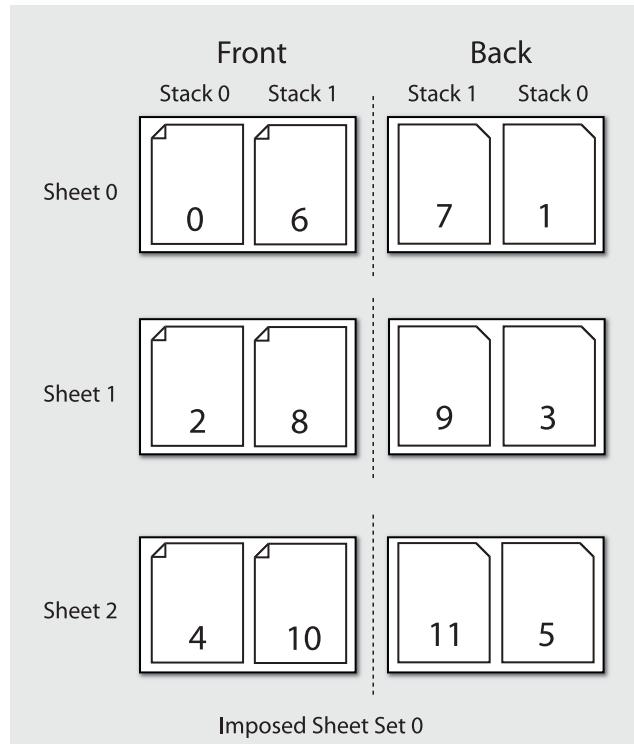
When insufficient number of pages remain to complete all **Logical Stacks** in an **Imposed Sheet Set**, the imposition engine SHALL distribute all content evenly across **Logical Stacks** in order to minimize the number of sheets in that **Imposed Sheet Set**, while still honoring any restrictions specified in **Layout**/**SheetCondition**, **LogicalStackParams**/**@Restrictions** or **Layout**/**PageCondition**.

6.4.12.4.1.1 Imposition for Cut and Stack

This example shows how to configure for cut and stack imposition. Cut and stack produces a sequence of **Imposed Sheet Sets**, where each **Imposed Sheet Set** is cut into separate physical stacks, then each physical stack is restacked into a larger stack. This simple example is configured for 2 **Logical Stacks** with a `@MaxStackDepth = "3"`, and is filled with 20 pages. Content on the back of the sheet is placed head-to-head with the front content.

Note: that the 2nd **Imposed Sheet Set** has distributed the remaining 8 pages onto 2 sheets.

Figure 6-1: Imposition for Cut and Stack



6.4.12.4.2 Imposition for Signatures with Saddle Stitching

Saddle stitched booklets typically contain pages selected from the front of the reader ordered list of pages and pages selected from the back of the reader ordered list of pages on the same sheet. For instance the outside cover of a 16 page booklet will contain the first page (*@Ord = "0"*) on the right of the sheet and the last page (*@Ord = "15"*) on the left of the sheet. The pagination for the inner sheets is calculated by adding to the page number from the front and by subtracting from the back. The next page inside the cover of a booklet printed in duplex will typically contain the third page (*@Ord = "2"*) on the right and the third from last page (*@Ord = "13"*) on the left. This behavior is described by specifying negative *@Ord* values for the **ContentObject** Elements that are filled with pages from the back of the **RunList** in automated imposition. The following code illustrates how absolute *@Ord* values are assigned based on sheet iterations.

Note: **Layout/@MaxCollect** specifies the maximum number of sheets per signature (e.g., in a perfect bound book). **@MaxCollect** specifies the maximum number of loops prior to restarting the signature.

Example 6-7: Automated Imposition: Ord Values

TBD 2.x Example.

```
/*
 * calculates a "real" ord value in an automated layout
 *
 * @param ord the Value of Ord in the layout
 * @param nPages the total number of pages that are consumed by the Layout, if
 *   frontOffset!=0 the pages before frontOffset are NOT counted
 * @param loop which sheet loop are we on?
 * @param maxOrdFront number of pages consumed from the front of the list
 * @param maxOrdBack positive number of pages consumed from the back of the list
 * @param frontOffset page number of the first page to be placed on ord 0 in loop 0
 * @return the pge to assign in this Ord, -1 if no page fits
 */

public static int calcOrd(int ord, int nPages, int loop, int maxOrdFront,
    int maxOrdBack, int frontOffset){
    final int maxOrd = maxOrdFront + maxOrdBack;
    if(maxOrd*loop >= nPages){
        return -1; // we are in a loop that has no remaining pages
    }
    int page;
    if(ord >= 0){ // count from front
        page = ord + loop*maxOrdFront;
    } else { // the page to put on -1
        int end = nPages + maxOrd - 1 - ((nPages +maxOrd - 1)%maxOrd);
        page = end - loop*maxOrdBack+ord;
    }
    // if a page evaluates to e.g. 10 and we only have 9 pages, ciao
    return page< nPages? page+frontOffset : -1;
}
```

6.4.12.4.3 Imposition for Start of a Chapter

The **Layout/PageCondition** Element may be used to specify where on a sheet a first page of a chapter (**Page Pool**) starts. It does this by specifying which **ContentObject** Elements on a sheet may not be used to place the first page of a chapter. An example may be found after Table 8-162, “PageCondition Element” on page 734.

6.4.12.4.4 Imposition for Regenerating Sheet Surfaces

There are two methods to configure the imposition engine for re-imposing sheet surfaces:

- 1 **Re-imposition by sheet or sheet surface:** A specific selection of sheets or surfaces imposed by the imposition engine may be selected using the controls of the **ParameterSet/[@Name="RunList"]** to the **RunList** (*Surface*) output from the **Imposition** Process.

- 2 **Re-imposition of sheets from content:** Alternatively the ParameterSet/[@Name="RunList"] to the **RunList** (*Document*) input to the imposition engine may be Partitioned to select specific content to be re-imposed.

For example, if the *@Metadata0* Partition Key has been configured to represent a recipient record number in a VDP job, that Partition Keys can be used to select a specific recipient record(s) for which to re-impose sheet surfaces.

Details on how to configure Resource/Part elements for sheet re-imposition including how to correctly regenerate dynamic sheet marks may be found at Section 3.12, “ResourceSet” on page 84 and *@IgnoreContext* in Table 8-269, “RunList Resource” on page 889.

6.4.12.4.5 Imposition for Document-Major Processing of a VDP Structured PDL

To process a Structured PDL in Document Major Processing Order, the **RunList** (*Document*) input ResourceSet SHALL contain Resource/Part Elements specifying the order in which documents SHALL be processed. This effects a virtual reordering of the content present in the PDL. Details on how to configure Resource/Part Elements for content reordering may be found at Section 3.12, “ResourceSet” on page 84 and *@IgnoreContext* in Table 8-269, “RunList Resource” on page 889.

6.4.13 InkZoneCalculation

The **InkZoneCalculation** Process takes place in order to preset the ink zones before printing. The **Preview** data are used to calculate a coverage profile that represents the ink distribution along and perpendicular to the ink zones within the printable area of the preview. The **InkZoneProfile** can be combined with additional, vendor-specific data in order to preset the ink zones and the oscillating rollers of an offset printing press.

Table 6-37: InkZoneCalculation – Input Resources

Name	Description
InkZoneCalculationParams ?	Specific information about the printing press geometry (e.g., the number of zones) to calculate the InkZoneProfile .
Layout ?	Specific information about the Media (including type and color) and about the Sheet (placement coordinates on the printing cylinder).
Preview ?	A low to medium resolution bitmap file representing the content to be printed.
Generic Input Resources *	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-38: InkZoneCalculation – Output Resources

Name	Description
InkZoneProfile ?	Contains information about ink coverage along and perpendicular to the ink zones for a specific press geometry.

6.4.14 Interpreting

The interpreting Device consumes page descriptions and instructions for controlling the marking Device (e.g., imag-setter, digital printers, CTP, digital printing Combined Processes, etc.). The parsing of graphical content in the page descriptions produces a canonical display list of the elements to be drawn on each page.

The interpreter SHALL act upon any Device control instructions that affect the physical functioning of the marking Device such as media selection and page delivery and implied **ColorSpaceConversion**. **Media** selection determines which type of medium is used for printing and where that medium can be obtained. Page delivery controls the location, orientation and quantity of physical output.

The interpreter is also responsible for resolving all system Resource references. This includes handling font substitutions and dealing with Resource aliases. However, the interpreter specifically does not get involved with any functions of the Device that could be considered finishing features such as stapling, duplexing and collating.

Table 6-39: Interpreting – Input Resources

Name	Description
ColorantControl ?	Identifies the color model used by the Job.
FontPolicy ?	Describes the behavior of the font machinery in absence of requested fonts.
InterpretingParams ?	Provides the parameters needed to interpret the PDL pages specified in the RunList Resource.
RunList ?	This Resource identifies a set of PDL pages or surfaces which will be interpreted.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-40: Interpreting – Output Resources

Name	Description
RunList ?	Pipe of streamed data which represents the results of <i>Interpreting</i> the pages in the RunList . In general, it is assumed that the <i>Interpreting</i> and <i>Rendering</i> Processes are tightly coupled and that there is no value in attempting to develop a general specification for the format of this data.

6.4.15 LayoutElementProduction

This Process describes the creation of page elements. It also explains how to create a layout that can put together all of the necessary page elements, including text, bitmap images, vector graphics, PDL or application files such as Adobe InDesign®, Adobe PageMaker® and Quark XPress®. The elements might be produced using any of a number of various software tools. This Process is often performed several times in a row before the final **RunList**, representing a final page layout file, is produced.

Table 6-41: LayoutElementProduction – Input Resources

Name	Description
LayoutElementProductionParams ?	The parameters for the <i>LayoutElementProduction</i> Process.
RunList ?	Location or metadata about the PDL or application file, bitmap image file, text file, vector graphics file, etc.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-42: LayoutElementProduction – Output Resources

Name	Description
RunList ?	A RunList is produced.

6.4.16 LayoutShifting

LayoutShifting specifies how to apply separation dependent shifts on a flat or objects on a press sheet.

The exact ordering of the process within the *Interpreting*, *Rendering* and *ImageSetting* and the Elements referenced by Input and Output **RunList** Elements are not defined. *LayoutShifting* MAY occur on display lists, raster data or in the image setting Hardware.

Table 6-43: LayoutShifting – Input Resources (Sheet 1 of 2)

Name	Description
LayoutShift ?	Parameters for the <i>LayoutShifting</i>

Table 6-43: LayoutShifting – Input Resources (Sheet 2 of 2)

Name	Description
RunList ?	References the input objects/flats to apply shifting to.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-44: LayoutShifting – Output Resources

Name	Description
RunList ?	The output RunList references the image data that the separation dependent layout shifts applied to.

6.4.17 PDLCreation

The **PDLCreation** Device consumes the display list of graphical elements generated by an **Interpreting**, **RasterReading** or a **ByteMap** and produces a new PDL output **RunList** based on the selected Output Resource parameters.

Table 6-45: PDLCreation – Input Resources

Name	Description
ImageCompressionParams ?	This Resource provides a set of controls that determines how images will be compressed in the resulting PDL pages.
PDLCreationParams ?	These parameters control the operation of the Process that interprets the display list and produces the resulting PDL pages.
RunList ?	This Resource is a Pipe of streamed data that represents a Device independent display list structure. The RunList SHALL specify a ByteMap Element.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-46: PDLCreation – Output Resources

Name	Description
RunList ?	This Resource identifies the location of the resulting PDL file(s). If the FileSpec/@MimeType is specified, then the value SHALL match PDLCreationParams/@MimeType . If not specified, then PDLCreationParams/@MimeType is inserted.

6.4.18 Preflight

Preflighting is the process of examining the components of a print Job to ensure that the Job will print successfully and with the expected results. Preflight checks can be performed on each document or finished page identified within the associated **RunList** Resource.

Preflighting a file is generally a two-step process. First, the documents are analyzed and compared to the set of tests. Then, a preflight report is built to list the encountered issues (according to the tests).

Table 6-47: Preflight – Input Resources

Name	Description
PreflightParams ?	A specified list of tests against which documents and/or pages are to be tested.
RunList ?	The list of documents and/or pages to be preflighted.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-48: Preflight – Output Resources

Name	Description
PreflightReport ?	PreflightReport is a container for logging information that is generated by the Preflight Process.
RunList ?	The list of output documents that were repaired by the preflight process.

6.4.19 PreviewGeneration

The **PreviewGeneration** Process produces a low resolution **Preview** of each separation that will be printed. The **Preview** can be used in later Processes such as **InkZoneCalculation**. The **PreviewGeneration** Process typically takes place after **Imposition**, **Interpreting**, **Rendering** or **RIPing**.^{####}

The **PreviewGeneration** can be performed in one of the following two ways: 1) the imaged printing plate is scanned by a conventional plate scanner or 2) medium to high resolution digital data are used to generate the **Preview** for the separation(s). The extent of the PDL coordinate system (as specified by the **MediaBox** Attribute, the resolution of the preview image, and width and height of the image) SHALL fulfill the following requirements:

$$\text{MediaBox-length} / 72 * \text{x-resolution} = \text{width} \pm 1$$

$$\text{MediaBox-height} / 72 * \text{y-resolution} = \text{height} \pm 1$$

A gray value of 0 represents full ink, while a value of 255 represents no ink (see the DeviceGray color model in [PS] Chapter 4.8.2).

Rules for the Generation of the Preview Image

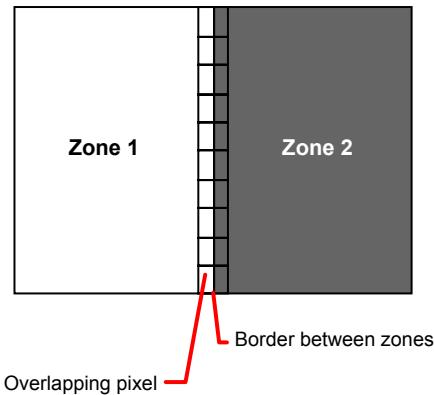
To be useful for the ink consumption calculation, the preview data SHALL be generated with an appropriate resolution. This means not only spatial resolution, but also color or tonal resolution. Spatial resolution is important for thin lines, while tonal resolution becomes important with large areas filled with a certain tonal value. The maximum error caused by limited spatial and tonal resolution SHOULD be less than 1%.

Spatial Resolution

Since some pixels of the preview image might fall on the border between two zones, their tonal values SHALL be split up. In a worst case scenario, the pixels fall just in the middle between a totally white and a totally black zone. In this case, the tonal value is 50%, but only 25% contributes to the black zone. With the resolution of the preview image and the zone width as variables, the maximum error can be calculated using the following equation:

$$\text{error}[\%] = \frac{100}{4 * \text{resolution}[L/mm] * \text{zone_width}[mm]}$$

For zone width broader than 25 mm, a resolution of 2 lines per mm will always result in an error less than 0.5%. Therefore, a resolution of 2 lines per mm (equal to 50.8 dpi) is suggested.

Figure 6-2: Worst case scenario for area coverage calculation

Tonal Resolution

The kind of error caused by color quantization depends on the number of shades available. If the real tonal value is rounded to the closest (lower or higher) available shade, the error can be calculated using the following equation:

$$\text{error}[\%] = \frac{100}{2 * \text{number_of_shades}}$$

Therefore, at least 64 shades SHOULD be used.

Line Art Resolution

When rasterizing line art elements, the minimal line width is 1 pixel, which means 1/resolution. Therefore, the relationship between the printing resolution and the (spatial) resolution of the preview image is important for these kind of elements. In addition, a specific characteristic of PostScript RIPs adds another error: within PostScript, each pixel that is touched by a line is set. Tests with different PostScript Jobs have shown that a line art resolution of more than 300 dpi is normally sufficient for ink-consumption calculation.

Conclusion

There are quite a few different ways to meet the requirements listed above. The following list includes several examples:

- The Job can be RIPed with 406.4 dpi monochrome.
- With anti-aliasing, the image data can be filtered down by a factor of 8 in both directions. This results in an image of 50.8 dpi with 65 color shades.
- High resolution data can also be filtered using anti-aliasing. First, the RIPed data, at 2540 dpi monochrome, are taken and filtered down by a factor of 50 in both directions. This produces an image of 50.8 dpi with 2501 color shades. Finally those shades are mapped to 256 shades, without affecting the spatial resolution.

Rasterizing a Job with 50.8 dpi and 256 shades of gray is not sufficient. The problem in this case is the rendering of thin lines (see Line Art Resolution above).

Recommendations for Implementation

The following three guidelines are strongly RECOMMENDED:

- The resolution of RIPed line art SHOULD be at least 300 dpi.
- The spatial resolution of the preview image SHOULD be approximately 20 pixel/cm (= 50.8 dpi).
- The tonal resolution of the preview image SHOULD be at least 64 shades.

Table 6-49: PreviewGeneration – Input Resources

Name	Description
ColorantControl ?	The ColorantControl Resources that define the ordering and usage of inks in print modules. Needed for generating thumbnails.
ExposedMedia ?	The PreviewGeneration Process can use an exposed printing plate to produce a Preview Resource. This task is performed using an analog plate-scanner. Exactly one of ExposedMedia , Preview or RunList SHALL be specified in any PreviewGeneration Process.
Preview ?	Medium or low resolution bitmap file that can be used for calculation of overviews and thumbnails. Exactly one of ExposedMedia , Preview or RunList SHALL be specified in any PreviewGeneration Process.
PreviewGenerationParams ?	Parameters specifying the size and the type of the preview.
RunList ?	High resolution bitmap data are consumed by the PreviewGeneration Process. These data represent the content of a separation that is recorded on a printing plate or other such item. Exactly one of ExposedMedia , Preview or RunList SHALL be specified in any PreviewGeneration Process.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-50: PreviewGeneration – Output Resources

Name	Description
Preview ?	The Preview data are comprised of low to medium resolution bitmap files representing, for example, the content of a separation that is recorded on a printing plate or other such item. A Preview can also be used to visualize Resources, such as thumbnail images.

6.4.20 RasterReading

The **RasterReading** Device consumes raster graphic formatted files into a display list structure as the principal element to be drawn on each page. The **RasterReading** Process is not a stand-alone Process but is used in conjunction with processing and rendering Processes in a Combined Process such as **Rendering** or **PDLCreation**.

Table 6-51: RasterReading – Input Resources

Name	Description
RasterReadingParams ?	Additional parameters for reading raster files.
RunList ?	This Resource identifies a set of raster pages or surfaces that will be inserted into the display list. This Resource SHALL reference ByteMap images.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-52: RasterReading – Output Resources

Name	Description
RunList ?	Pipe of streamed data that represents the results of RasterReading the pages in the input RunList . The format and detail are implementation dependent. The RunList SHALL specify the output content data for RasterReading .

6.4.21 Rendering

The **Rendering** Process consumes the display list of graphical elements generated by the **Interpreting** or **RasterReading** Process. It converts the graphical elements according to the geometric and graphic state information contained within the display list, combined with the **RenderingParams** information to produce binary rasterized data suitable for Processes which consume **ByteMap** information.

Table 6-53: Rendering – Input Resources

Name	Description
ImageCompressionParams ?	Allows definition of compressed Raster Images
RenderingParams ?	This Resource describes the format of the byte maps to be created and other specifics of the Rendering Process.
RunList ?	Pipe of streamed data that represents the results of Interpreting or RasterReading the pages in the input RunList . In general, it is assumed that the Interpreting , RasterReading , Rendering and PDLCreation are tightly coupled and that there is no value in attempting to develop a general specification for the format of this data.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-54: Rendering – Output Resources

Name	Description
RunList ?	Pipe of streamed data that represents the results of Rendering . This RunList MAY be consumed by any following Process that consumes raster data, including PDLCreation , ImageSetting or DigitalPrinting . The data MAY be specified in ByteMap sub-elements. In general, it is assumed that the Interpreting , RasterReading , Rendering and PDLCreation are tightly coupled and that there is no value in attempting to develop a general specification for the format of this data.

```
<JDF Type="ProcessGroup" Types="RIPing" Category="RIPing"
      ID="ID100" JobPartID="ID23" Status="Ready" Version="1.4" />
```

6.4.22 Screening

This Process specifies the Process of halftone screening. It consumes contone raster data (e.g., the output from an **Interpreting** and **Rendering** Process). It produces monochrome which has been filtered through a halftone screen to identify which pixels are needed to approximate the original shades of color in the document.

This Process definition includes capabilities for post-RIP halftoning according to the PostScript definitions. Alternatively it allows for the selection of FM screening/error diffusion techniques. In general, an actual screening Process will be a Combined Process of **ContoneCalibration** and **Screening** Processes.

Table 6-55: Screening – Input Resources (Sheet 1 of 2)

Name	Description
RunList ?	Ordered list of rasterized ByteMap or interpreted data representing pages or surfaces.
ScreeningParams ?	Parameters specifying which halftone mechanism is to be applied and with what specific controls.

Table 6-55: Screening – Input Resources (Sheet 2 of 2)

Name	Description
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-56: Screening – Output Resources

Name	Description
RunList ?	Ordered list of rasterized and screened output pages. Assumes that the resolution remains the same and that resulting data are one bit per component. Furthermore, the organization of planes within the data does not change.

6.4.23 Separation

The **Separation** Process specifies the controls associated with the generation of color-separated data. **Separation** may be applied either to pdl data or raster data.

Table 6-57: Separation – Input Resources

Name	Description
ColorantControl ?	Identifies which colorants in the Job are to be output.
RunList ?	List of elements, surfaces or pages that are to be operated on.
SeparationControlParams ?	Controls for the separation Process.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-58: Separation – Output Resources

Name	Description
RunList	List of separated elements, separated surfaces, separated pages or separated raster bytemaps.

6.4.24 SheetOptimizing

SheetOptimizing describes ganging of multiple sections onto one or more printed sheets. Sections MAY be parts of unrelated customer jobs. This process is also referred to as job ganging.

SheetOptimizing MAY be used together with QueueSubmissionParams/@*GangName* and the ForceGang Command. In this case, individual jobs with identical QueueSubmissionParams/@*GangName* are collected with each job submission. A ForceGang Command instructs the Ganging engine to process the waiting GangInfo Elements.

Table 6-59: SheetOptimizing – Input Resources (Sheet 1 of 2)

Name	Description
Assembly *	Input assemblies to specify the binding order for creep calculation. These Assemblies MAY contain sections that are not included in this sheet optimization (e.g., when only covers are optimized and the bodies are produced individually).
SheetOptimizingParams ?	Parameters specifying details that allow individual sections to be distributed on the printed sheets.

Table 6-59: SheetOptimizing – Input Resources (Sheet 2 of 2)

Name	Description
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-60: SheetOptimizing – Output Resources

Name	Description
Layout	The Layout Resource that will be populated by the SheetOptimizing Process. The Resource MAY be partially populated by the submitter with restrictions on what the SheetOptimizing is allowed to do.

6.4.25 Stripping

An important aspect of the interface between an MIS system and a prepress workflow system is imposition. When an order is accepted or even during the estimation phase, the MIS system determines how the product will be produced using the available equipment (e.g., presses, folders, cutters, etc.) in the most cost-efficient way. The result of this exercise has a large impact on imposition in prepress.

The **Stripping** Process specifies the Process of translating a high level structured description of the imposition of one or multiple Job Parts or part versions represented by a partially populated **Layout** Resource into a fully populated **Layout** Resource for the **Imposition** Process. Note that the **Stripping** Process can generate all Resources needed for the **Imposition** Process, thus also the **RunList** (*Marks*).

The **BinderySignature** elements that are referenced from **Layout/@BinderySignatureRef** define the individual Signatures that are combined to produce a final product. The **Assembly** specifies how the individual **BinderySignature** elements are combined relative to one another.

6.4.25.1 Pagination in Stripping

The distribution and orientation of pages on a **BinderySignature** is determined by the geometry of the final product. The **Assembly** Resource determines which pages SHALL be placed on which **BinderySignature**.

Example: if two 8 page **BinderySignatures** are gathered on top of one another, then pages 1-8 will go on the first **BinderySignature** and pages 9-16 will go on the second **BinderySignature**. If the same **BinderySignatures** are collected on a saddle, then pages 1-4 and 13-16 will go on the first **BinderySignature** and pages 5-12 will go on the second **BinderySignature**.

The **BinderySignature** determines how the pages that are selected by the **Assembly** SHALL be distributed on each **BinderySignature**. The page distribution is modified by the following attributes in **BinderySignature**:

- **@BinderySignatureType** determines whether the pagination SHALL be explicitly defined or SHALL be calculated from **@FoldCatalog**,
- **@BindingEdge** and **@JogEdge** using the methods defined in Appendix J, “Pagination Catalog” on page 1207.

Table 6-61: Stripping – Input Resources (Sheet 1 of 2)

Name	Description
Assembly *	Assembly describes how the BinderySignatures are combined to a final product. Multiple Assemblies are typically specified when multiple jobs are ganged on a sheet.
ColorantControl ?	Contains information on the colors and separations. Useful when creating marks that need color information.
Layout ?	High level structured description of the imposition of one or multiple fold sheets.

Table 6-61: Stripping – Input Resources (Sheet 2 of 2)

Name	Description
RunList (Document) ?	List of documents. When available, this list can be used to generate a Layout and populated RunList (no RunList[@ElementType = "Reservation"]) which can be fed into a subsequent Imposition Process.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-62: Stripping – Output Resources

Name	Description
Layout ?	The detailed layout of the pages to be imposed.
RunList (Document) ?	List of document pages that SHALL be used as input of the following Imposition Process.
RunList (Marks) ?	List of marks that SHALL be used as input of the following Imposition Process.

Example 6-8: Stripping: Simple Example

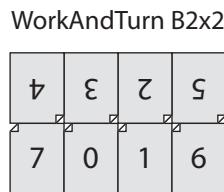
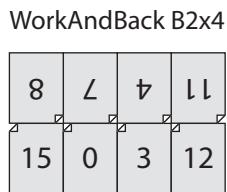
The first example specifies three Sheets based on folding catalog example F16-6. More examples can be found in Section O.6, “Stripping”.

TBD 2.x Example.

```
<StrippingParams ID="FoldCatalogSample" Class="Parameter" Status="Available"
  WorkStyle="WorkAndBack" PartIDKeys="SheetName">
  <BinderySignature FoldCatalog="F16-6"/>
  <StrippingParams SheetName="Sheet1"/>
  <StrippingParams SheetName="Sheet2"/>
  <StrippingParams SheetName="Sheet3"/>
</StrippingParams>
```

Example 6-9: Stripping: Complex Example

The following example specifies three Sheets: *Sheet1* and *Sheet2* are based on a *B2x4 BinderySignature* using the “*WorkAndBack*” WorkStyle, while *Sheet3* is based on *BinderySignature B2x2* using the “*WorkAndTurn*” WorkStyle.



TBD 2.x Example.

```
<BinderySignature ID="B2x4" Class="Parameter" Status="Available"
  NumberUp="4 2">
  <SignatureCell FrontPages="15 0 3 12" BackPages="14 1 2 13"
    Orientation="Up"/>
  <SignatureCell FrontPages="8 7 4 11" BackPages="9 6 5 10"
    Orientation="Down"/>
</BinderySignature>
<BinderySignature ID="B2x2" Class="Parameter" Status="Available"
  NumberUp="2 2">
```

```

<SignatureCell FrontPages="7 0" BackPages="6 1" Orientation="Up"/>
<SignatureCell FrontPages="4 3" BackPages="5 2" Orientation="Down"/>
</BinderySignature>
<StrippingParams ID="L1" Class="Parameter" Status="Available"
    WorkStyle="WorkAndBack" PartIDKeys="SheetName">
    <StrippingParams SheetName="Sheet1">
        <BinderySignatureRef rRef="B2x4"/>
    </StrippingParams>
    <StrippingParams SheetName="Sheet2">
        <BinderySignatureRef rRef="B2x4"/>
    </StrippingParams>
    <StrippingParams WorkStyle="WorkAndTurn" SheetName="Sheet3">
        <BinderySignatureRef rRef="B2x2"/>
        <Position RelativeBox="0 0 0.5 1"/>
        <Position RelativeBox="0.5 0 1 1" Orientation="Flip180"/>
    </StrippingParams>
</StrippingParams>

```

6.4.26 Tiling

The **Tiling** Process allows the contents of Surfaces to be imaged onto separate pieces of media. Note that many different workflows are possible. **Tiling** SHALL always follow **Imposition**, but it can operate on imposed PDL page contents or on contone or halftone data. **Tiling** will generally be part of a Combined Process. For example, **Tiling** might be part of a Combined Process with **ImageSetting**. In that case, the input would be a **RunList** that contains ByteMap Resources for each surface.

Table 6-63: Tiling – Input Resources

Name	Description
RunList (Marks) ?	Structured list of incoming marks. These are typically printer's marks that provide the information needed to combine the tiles.
RunList (Surface) ?	Structured list of imposed page contents or Byte Maps that are to be decomposed to produce the images for each tile. The @ElementType value of the RunList Resource SHALL be " <i>Surface</i> ".
Tile ?	A Partitioned Tile Resource that describes how the surface contents are to be decomposed.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-64: Tiling – Output Resources

Name	Description
RunList ?	Structured list of portions of the decomposed surfaces. The value of the @ElementType Attribute of the RunList element SHALL be " <i>Tile</i> ".

6.4.27 Trapping

Trapping is a prepress Process that modifies PDL files to compensate for a type of error that occurs on presses. Specifically, when more than one colorant is applied to a piece of media using more than one inking station, the media might not stay in perfect alignment when moving between inking stations. Any misalignment will result in an error called misregistration. The visual effect of this error is either that inks are erroneously layered on top of one another, or, more seriously, that gaps occur between inks that are intended to abut. In this second case, the color of the media is revealed in the gap and is frequently quite noticeable. **Trapping**, in short, is the Process of modifying PDL files so that abutting colorant edges intentionally overlap slightly, in order to reduce the risk of gaps.

The **Trapping** Process modifies a set of document pages to reduce or (ideally) eliminate visible misregistration errors in the final printed output.

Table 6-65: Trapping – Input Resources

Name	Description
ColorantControl ?	Identifies color model used by the Job.
FontPolicy ?	Describes the behavior of the font machinery in absence of requested fonts.
RunList ?	Structured list of incoming page contents that are to be trapped.
TrappingParams ?	Describes the general setting needed to perform trapping.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-66: Trapping – Output Resources

Name	Description
RunList ?	Structured list of the modified page contents after Trapping has been executed.

6.5 Press Processes

Press Processes are various technological procedures involving the transfer of ink to a substrate. From a technical standpoint they are often classified in impact and non-impact printing technologies. The impact printing class can be further subdivided into relief, intaglio, planograph or screen technologies, which in turn can be divided in further sub-parts. Because of the way a workflow is constructed in XJDF, however, a different approach to classification was used. All of the various printing technologies are gathered into two categories: 1) **ConventionalPrinting**, which involves printing from a physical master, 2) **DigitalPrinting**, which involves generic commercial printing from a digital master.

The most prominent physical, planographic printing technologies are offset lithography and electrophotography. They are also the printing Processes with the highest adoption in today's graphic arts industry. Consequently, the **ConventionalPrinting** Process in XJDF takes them as models. That does not mean, however, that other printing techniques can not make use of the **ConventionalPrinting** Process and its Resources. The extensibility features of XJDF can be used to fill other requirements related to printing technology.

6.5.1 ConventionalPrinting

- This Process covers several conventional printing tasks, including Sheet-Fed printing, Web Printing, Web/ribbon coating, converting and varnishing. Typically, each takes place after prepress and before postpress Processes. Direct imaging technology on press is modelled as a Combined Process of **ImageSetting** and **ConventionalPrinting**. Press machinery often includes postpress Processes (e.g., **WebInlineFinishing**, **Folding** and **Cutting**) as in-line finishing operations. The **ConventionalPrinting** Process itself does not cover these postpress tasks. Using a conventional printing press for producing a pressproof can be performed formally by producing a proof of type **Component** with a **ConventionalPrinting** Process. The result of this Process is then sent to the **Approval** Process, which in turn produces an **ApprovalDetails** Resource. That Resource is then passed on to a second **ConventionalPrinting** Process, which requires that the press be set up a second time.

Note that the definition and ordering of separations is specified by the **@DeviceColorantOrder** Attribute of the appropriate **ColorantControl** Resource.

In the context of Web Printing, the **ConventionalPrinting** Process SHALL be in a Combined Process with the **WebInlineFinishing** Process. The following drawing gives an overview about Web Printing in general.

Figure 6-3: Overview of Web Printing

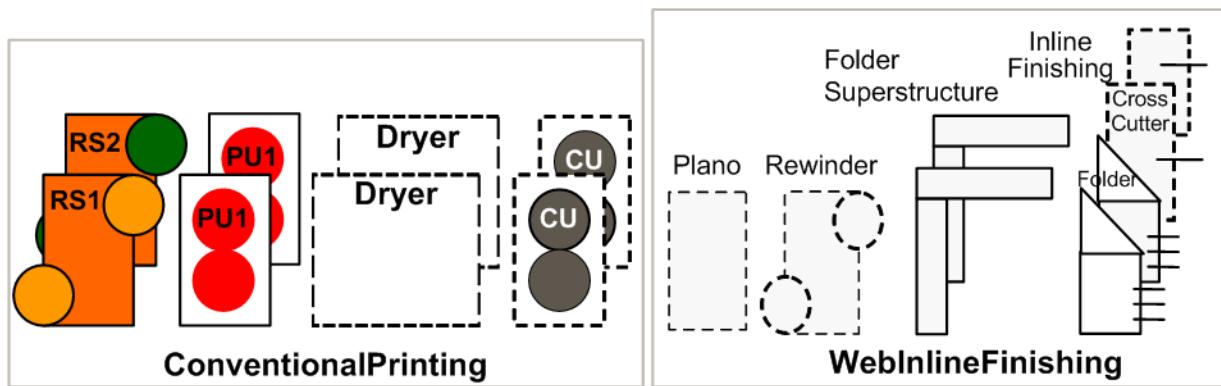


Table 6-67: ConventionalPrinting – Input Resources (Sheet 1 of 2)

Name	Description
ColorantControl ?	The ColorantControl Resources that define the ordering and usage of inks in print modules. The ColorantControl Resource specifies the complete set of colors that will be printed on a Sheet.
Component ?	Various components in the form of preprints can be used in ConventionalPrinting . The most common Component used is unprinted Paper.
ConventionalPrintingParams ?	Specific parameters to set up the press.
ExposedMedia (<i>Plate</i>) ?	The printing plates and information about them are used to set up the press. The ExposedMedia (<i>Plate</i>) Resource defines the set of plates to be used in the press run that is described by this Node. Both ExposedMedia (<i>Cylinder</i>) and ExposedMedia (<i>Plate</i>) MAY occur in the same Device. At least one of ExposedMedia (<i>Cylinder</i>) or ExposedMedia (<i>Plate</i>) SHALL be specified.
ExposedMedia (<i>Sleeve</i>) ?	Description of a sleeve.
Ink ?	Information about the ink (e.g., brand, color) is useful to set up the press.
InkZoneProfile ?	The InkZoneProfile contains information about how much ink is needed along the printing cylinder of a specific printing press. It is only useful for Offset Lithography presses with ink key adjustment functions.
Layout ?	Sheet and surface elements from the Layout tree (e.g., CIELABMeasuringField , DensityMeasuringField or ColorControlStrip) can be used for quality control at the press. The quality control field value and position can be of interest for automatic quality control systems. RegisterMark can be used to line up the printing plates for the press run, and its position can in turn be used to position items such as a camera.
Media (<i>MountingTape</i>) ?	Description of a mounting tape for a sleeve.
PrintCondition ?	Used to control the use of colorants when printing pages on a specific media. The Attributes and Elements of the PrintCondition Resource describe the aim values for a given printing Process.

Table 6-67: ConventionalPrinting – Input Resources (Sheet 2 of 2)

Name	Description
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-68: ConventionalPrinting – Output Resources

Name	Description
Component ?	Describes the printed Sheets, ribbons or webs which can be used by another printing Process or postpress Processes. Note that the <i>@Amount</i> Attribute of the <i>PartAmount</i> to this Resource indicates the number of copies of the entire Job which will be produced.

6.5.2 DigitalPrinting

DigitalPrinting is a direct printing Process that, like **ConventionalPrinting**, occurs after prepress Processes but before postpress Processes. In **DigitalPrinting**, the data to be printed are not stored on an extra medium (e.g., a printing plate or a printing foil), but instead are stored digitally. The printed image is generated for every output using the digital data. Electrophotography, inkjet, and other technologies are used for transferring ink (both liquid ink and dry toner) onto the substrate. Furthermore, both Sheet-Fed and Web Presses can be used as machinery for **DigitalPrinting**. The **DigitalPrinting** Process SHALL also be used to describe hard copy proofing (see Section 6.3.1, “Approval” on page 358).

DigitalPrinting is often used to image a small area on preprinted **Component** Resources to perform actions such as addressing or numbering another **Component**. This kind of Process can be executed by imaging with an inkjet printer during press, postpress or packaging operations. Therefore, **DigitalPrinting** is not only a press or pre-press operation but sometimes also a postpress Process.

Digital printing Devices which provide some degree of finishing capabilities (e.g., collating and stapling) as well as some automated layout capabilities (e.g., N-up and duplex printing) MAY be modeled as a Combined Process which includes **DigitalPrinting**. Such a Combined Process MAY also include other Processes (e.g., **Approval**, **ColorCorrection**, **ColorSpaceConversion**, **ContoneCalibration**, **Cutting**, **Folding**, **HoleMaking**, **Imposition**, **Interpreting**, **Perforating**, **Rendering**, **Screening**, **Stacking**, **Stitching**, **Trapping** or **Trimming**).

Controls for **DigitalPrinting** are provided in the **DigitalPrintingParams** Resource. The set of Input Resources of a Combined Process which includes **DigitalPrinting** MAY be used to represent an Internet Printing Protocol (IPP) Job or a PPML Job. See Application Notes for IPP and Variable Data printing. Note that putting a label on a product or **DropItem** is not **DigitalPrinting** but **Inserting**.

Table 6-69: DigitalPrinting – Input Resources (Sheet 1 of 2)

Name	Description
ColorantControl ?	The ColorantControl Resources that define the ordering and usage of inks in print modules.
Component *	Various components can be used in DigitalPrinting . Examples include unprinted paper, preprinted covers, waste, precut Media , or a set of preprinted Sheets or webs. If multiple Component (Input) Resources are linked to one Process, the mapping of media to content is defined in the Partitions of DigitalPrintingParams .
DigitalPrintingParams ?	Specific parameters to set up the machinery.
Ink ?	Ink or toner and information that is needed for DigitalPrinting .

Table 6-69: DigitalPrinting – Input Resources (Sheet 2 of 2)

Name	Description
Layout ?	Sheet and surface Elements from a Layout (e.g., the CIELABMeasuringField , DensityMeasuringField or ColorControlStrip) can be used for quality control at the press. The value and position of the quality can be of interest for automatic quality control systems. RegisterMark Resources can be used to line up the printing registration during press run, and its position can in turn be used to position an item such as a camera.
PrintCondition ?	Used to control the use of colorants when printing pages on a specific media. The Attributes and Elements of the PrintCondition Resource describe the aim values for a given printing Process.
RunList ?	Rendered data that will be printed on the digital press are needed for DigitalPrinting .
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-70: DigitalPrinting – Output Resources

Name	Description
Component ?	Components are produced for other printing Processes or postpress Processes. Note that ./Resource/AmountPool/PartAmount/@Amount indicates the number of copies of the entire Job which will be produced. @ProcessUsage = "Good" is OPTIONAL. Note: when processing a PDL with multiple documents or sets, such as pdf/vt, the amount is defined in the scope of the entire document. If one copy of the number of copies defined within the PDL file of each record is requested, the Component/@Amount SHALL be set to 1.

6.5.3 Varnishing

Varnishing is the Process of varnishing to a blank or printed sheet. Spot varnishing with a ripped image or a printing plate from **ExposedMedia** is described as **DigitalPrinting** or **ConventionalPrinting** with **Ink/@Family = "Varnish"**. All types of all-over (flood) Varnishing or Spot Varnishing applied without a ripped image or a printing plate from **ExposedMedia** are described with the **Varnishing** process.

Table 6-71: Varnishing – Input Resources

Name	Description
Component ?	The Component to be varnished.
ExposedMedia *	Various types of ExposedMedia MAY be specified for varnishing. See VarnishingParams/@VarnishMethod for details
Ink ?	Details of the colorant that is used for Varnishing . Ink/@Family SHOULD be "Varnish".
VarnishingParams ?	Details of the setup of the varnishing device
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-72: Varnishing – Output Resources

Name	Description
Component ?	The varnished Component .

6.6 Postpress Processes

In this specification, the postpress Processes are presented in two parts: an alphabetical list of Processes that is then followed by a Postpress Processes Structure section that divides these Processes into subchapters for structuring purposes. This structuring is useful to find specific Processes. Please note that Processes, in some cases can be used to describe operations that go beyond the scope of a specific chapter. Therefore, it is a good idea not only to look at certain Processes within a subchapter but also to find out what functionality other Processes offer if a specific task needs to be addressed.

6.6.1 BlockPreparation

As there are many options for a hardcover book, the block preparation is more complex than what has already been described for other types of binding. Those options are the ribbon band (numbers of bands, materials and colors), gauze (material and glue), headband (material and colors), kraft paper (material and glue) and tightbacking (different geometry and measurements).

Table 6-73: BlockPreparation – Input Resources

Name	Description
Component ?	The BlockPreparation Process consumes one Component and creates a book block.
BlockPreparationParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-74: BlockPreparation – Output Resources

Name	Description
Component ?	One Component is produced: the prepared book block. Its <i>@ProductType</i> = "BookBlock".

6.6.2 BoxFolding

BoxFolding defines the Process of folding and gluing blanks into folded flat boxes for packaging.

Table 6-75: BoxFolding – Input Resources

Name	Description
BoxFoldingParams ?	Specific parameters to set up the folder gluer.
Component	The BoxFolding Process consumes one Component , the folding blank. Its <i>@ProductType</i> = "BlankBox".
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-76: BoxFolding – Output Resources

Name	Description
Component ?	One Component is produced: the folded flat box. Its <i>@ProductType</i> = "FlatBox".

6.6.3 BoxPacking

A pile, stack or bundle of products can be packed into a box or carton.

Table 6-77: BoxPacking – Input Resources (Sheet 1 of 2)

Name	Description
BoxPackingParams ?	Specific parameters to set up the machinery.

Table 6-77: BoxPacking – Input Resources (Sheet 2 of 2)

Name	Description
Component *	The BoxPacking Process puts a set of Component Resources into the box Component . If more than one Component is specified, a Component/Bundle Resource SHALL also be specified for the output Component .
Component (Box) ?	Details of the box or carton.
Media (Tie) ?	Protective Media can be placed between individual rows of Component Resources.
Media (Underlay) ?	Protective Media can be placed between individual layers of Component Resources.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-78: BoxPacking – Output Resources

Name	Description
Component ?	One Component is produced: the Component that represents the packed Box.

6.6.4 Bundling

The **Bundling** Process normally will be followed by a **Strapping** Process. In a **Bundling** Process, single products like Sheets or Signatures are bundled. The bundle is the output **Component** of the Process and is used to store the products. As input a **Component** to a consuming or subsequent Process (e.g., **Gathering**, **Collecting** or **Inserting**), the single components of a bundle are used.

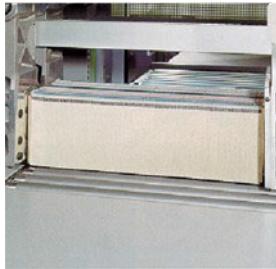
Figure 6-4: Bundle Creation**Figure 6-5: Bundle Transport**

Table 6-79: Bundling – Input Resources

Name	Description
BundlingParams ?	Bundling parameters.
Component ?	Component to be bundled
Media ?	End boards to protect the bundle. For each bundle a pair of end boards is needed.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-80: Bundling – Output Resources

Name	Description
Component ?	The completed bundle.

Parameters like manufacturer and Device type are defined in the **Device** Element.

6.6.5 CaseMaking

Case making is the Process where a hard cover book case is produced.

Table 6-81: CaseMaking – Input Resources

Name	Description
CaseMakingParams ?	Specific parameters to set up the machinery.
Component (CoverMaterial) ?	The cover material is either a preprinted or processed Sheet of paper.
Component (CoverBoard) ?	The cardboard Component used for the cover board.
Component (SpineBoard) ?	The cardboard Component used for the spine board. If not specified, the Component (CoverBoard) SHALL be used for the spine board.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-82: CaseMaking – Output Resources

Name	Description
Component ?	One Component is produced: the book case. Its <i>@ProductType = "BookCase"</i> .

6.6.6 CasingIn

The hard cover book case and the book block are joined in the **CasingIn** Process.

Table 6-83: CasingIn – Input Resources

Name	Description
CasingInParams ?	Specific parameters to set up the machinery.
Component ?	The prepared book block.
Component (Case) ?	The hard cover book case.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-84: CasingIn – Output Resources

Name	Description
Component ?	One Component is produced: the completed hard cover book.

6.6.7 Collecting

This Process collects folded Sheets or partial products, some of which might have been cut. The first **Component** to enter the workflow lies at the bottom of the pile collected on a saddle, and the sequence of the input components that follows depends upon the produced component. The figure to the right shows a typical collected pile.



The operation coordinate system is defined as follows: The y-axis is aligned with the binding edge. It increases from the registered edge to the edge opposite to the registered edge. The x-axis is aligned with the registered edge. It increases from the binding edge to the edge opposite to the binding edge (i.e., the product front edge).

Table 6-85: Collecting – Input Resources

Name	Description
Assembly ?	Explicitly describes the sequence of the Component Resources to be collected. If Assembly is not specified, the sequence is defined by the sequence of the Component . Caution: Assembly has the first on the outside, whereas the Component Resources are listed from inside to outside.
CollectingParams ?	Specific parameters to set up the machinery.
Component *	Variable amount of Sheets to be collected.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-86: Collecting – Output Resources

Name	Description
Component ?	A block of collected Sheets is produced. This Component can be joined in further postpress Processes.

6.6.8 CoverApplication

CoverApplication describes the Process of applying a soft cover to a book block.

Table 6-87: CoverApplication – Input Resources

Name	Description
Component ?	The book block on which the cover is applied. If Component (<i>Cover</i>) is NOT provided, this Component SHALL be Partitioned, and the first Partition of this Component SHALL specify the Cover.
Component (<i>Cover</i>) ?	The soft cover that is applied.
CoverApplicationParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-88: CoverApplication – Output Resources

Name	Description
Component ?	The book block with the applied soft cover.

6.6.9 Creasing

Sheets are creased or grooved to enable folding or to create even, finished page delimiters.

Table 6-89: Creasing – Input Resources

Name	Description
Component ?	This Process consumes one Component : the printed Sheets.
CreasingParams ?	Details of the <i>Creasing</i> Process.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-90: Creasing – Output Resources

Name	Description
Component ?	One creased Component is produced.

6.6.10 Cutting

Sheets are cut using a guillotine *Cutting* Machine. *CutBlock* Resources MAY be used for positioning the knife.

Since *Cutting* is described here in a way that is Machine independent as much as possible, the specified *CutBlock* Elements do not directly imply a particular cutting sequence. Instead, the Device SHALL determine the sequence.

Cutting MAY also be used to describe cutting of a web into multiple Ribbons on a web press. This process is commonly referred to as “Slitting”.

Table 6-91: Cutting – Input Resources

Name	Description
Component ?	This Process consumes one Component : the printed Sheets.
CuttingParams ?	Details of the <i>Cutting</i> Process.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-92: Cutting – Output Resources

Name	Description
Component *	One or several blocks of cut Component Resources are produced. When an input Component is cut, the output SHALL be a set of Component Resources.

6.6.11 DieMaking

This Process describes the production of Tools for a die cutter (e.g., in a die maker shop).

Table 6-93: DieMaking – Input Resources

Name	Description
DieLayout ?	A Resource describing the die cutter tool set.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types

Table 6-94: DieMaking – Output Resources

Name	Description
Tool *	The set of tools for the die cutter.

6.6.12 Embossing

The **Embossing** Process is performed after printing to stamp a raised or depressed image (artwork or typography) into the surface of paper using engraved metal embossing dies, extreme pressure and heat. Embossing styles include blind, deboss and foil-embossed.

Table 6-95: Embossing – Input Resources

Name	Description
Component ?	This Process consumes one Component which is embossed by the Process.
EmbossingParams ?	Parameters to setup the machinery.
Media (Foil) *	Media(Foil) SHOULD be provided if an EmbossingParams /Emboss/ @ <i>EmbossingType</i> = <i>FoilEmbossing</i> or <i>FoilStamping</i> .
Tool *	The embossing stamps or calenders.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-96: Embossing – Output Resources

Name	Description
Component ?	One Component is created.

6.6.13 EndSheetGluing

EndSheetGluing finalizes the folded Sheet or book block in preparation for case binding. It requires three **Component** Resources – the back-end Sheet, the book block and the front-end Sheet – and information about how they are merged together. Back-end Sheets and front-end Sheets are in most cases Sheets folded once before **EndSheetGluing** takes place. The end Sheets serve as connections between the book block and the cover boards.

Table 6-97: EndSheetGluing – Input Resources

Name	Description
Component ?	A back-end Sheet and a front-end Sheet are glued onto the book block. At least one of Component , Component (BackEndSheet) or Component (FrontEndSheet) SHALL be present.
Component (BackEndSheet) ?	A back-end Sheet to be mounted on the book block.
Component (FrontEndSheet) ?	A front-end Sheet to be mounted on the book block.
EndSheetGluingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-98: EndSheetGluing – Output Resources

Name	Description
Component ?	A book block is produced that includes the end Sheets.

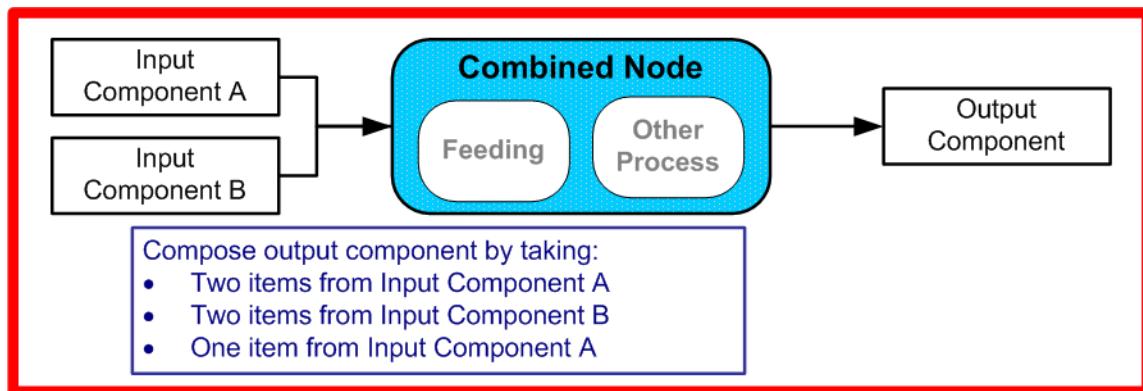
6.6.14 Feeding

The **Feeding** Process separates sheets or signatures from a stack or stream and feeds single **Component**(s) to Processes such as **Folding**, **Gathering**, **Collecting**, **ConventionalPrinting**, etc. In general, the **Feeding** Process will be part of a Combined Process with Processes that consume the feed of **Component**(s) or **Media**.

When used in a Combined Process with feed consuming Process (e.g., **Gathering**), the **Feeding** Process allows an arbitrary complex selection of input **Component** Elements in any number, and in any order, as long as elements are consumed consecutively (i.e., no random access within a single input component).

When specified for a web press or web finishing device, **Feeding** describes the process of unwinding **Media** or **Components** from a roll.

Figure 6-6: Combined Process with Feeding Process



In our example above, one input component (Component A) is a bundle component (`@BundleType = "Stack"`) consisting of a collated set of three Sheets, the other one (Component B) is a collated set consisting of two Sheets per set. Both sets are oriented face-up (See Figure 6-7). Figure 6-8 shows the output for the case of **Gathering**.

Figure 6-7: Input Components

- Sheet 1 of Component A — — Sheet 1 of Component B —
- Sheet 2 of Component A — — Sheet 2 of Component B —
- Sheet 3 of Component A —

Figure 6-8: Output Component

- Sheet 3 of Component A —
- Sheet 2 of Component B —
- Sheet 1 of Component B —
- Sheet 2 of Component A —
- Sheet 1 of Component A —

Note that, by default, none of the Sheets is flipped, so surfaces of Sheet 1 of **Component A** do not show in a different direction. To flip Sheets, **FeedingParams/CollatingItem/@Orientation** MAY be specified.

Table 6-99: Feeding – Input Resources (Sheet 1 of 2)

Name	Description
Component *	Sheets or Signatures to be fed to the machinery. The <code>@ProcessUsage</code> of the Component MAY be specified as any valid <code>@ProcessUsage</code> of the a feed consuming Process.

Table 6-99: Feeding – Input Resources (Sheet 2 of 2)

Name	Description
FeedingParams ?	Specific parameters to set up the Feeding Process
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-100: Feeding – Output Resources

Name	Description
Component *	Component(s) fed to the consuming Process.

6.6.15 Folding

Buckle folders or knife folders are used for **Folding** Sheets. One or more Sheets can be folded at the same time. Web presses often provide in-line **Folding** equipment. Longitudinal **Folding** is often performed using a former, a plow folder or a belt. While jaw folding, chopper folding or drum folding equipment is used for folding the Sheets that have been divided.

The **XJDF Folding** Process covers both operations done in stand-alone **Folding** machinery – typically found for processing printed materials from Sheet-Fed presses – and in-line equipment of Web Presses. Creasing and/or slot perforating are sometimes necessary parts of the **Folding** operation that guarantee exact Process execution. They depend on the folder used, the **Media** and the folding layout. These operations are specified in the **Creasing** and **Perforating** Processes respectively.

Table 6-101: Folding – Input Resources

Name	Description
Component ?	Component Resources, including a printed Sheet or a pile of Sheets, are used in the Folding Process.
FoldingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-102: Folding – Output Resources

Name	Description
Component ?	The Process produces a Component , which in most cases is a folded Sheet.

6.6.16 Gathering

In the **Gathering** Process, ribbons, Sheets or other **Component** Resources are accumulated on a pile that will eventually be stitched or glued in some way to create an individual **Component**. The input **Component** Resources MAY be Output Resources of a Web-Printing Machine used in **Collecting** or of any Machine that executes a **ConventionalPrinting** or **DigitalPrinting** Process. In Sheet applications, a moving gathering channel is used to transport the pile. But no matter what the inception of the **Gathering** Process, the sequence of the input components dictates the produced component. Figure 6-9, “Gathering,” on page 417 shows typical gathered piles.

Figure 6-9: Gathering

Table 6-103: Gathering – Input Resources

Name	Description
Assembly ?	Explicitly describes the sequence of the Component Resources to be gathered. If Assembly is not specified, the sequence is defined by the sequence of the Component . Caution: Assembly has the first on the top, whereas the Component Resources are listed from bottom to top.
Component*	Variable amount of components including single Sheets or folded Sheets are used in the Gathering Process. The first Component in the list lies at the bottom of the gathered pile.
GatheringParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-104: Gathering – Output Resources

Name	Description
Component ?	Components gathered together (e.g., a pile of folded Sheets).

6.6.17 Gluing

Gluing describes arbitrary methods of applying glue to a **Component**.

Table 6-105: Gluing – Input Resources

Name	Description
Component ?	This Process consumes one Component : the printed Sheets.
GluingParams ?	Details of the Gluing Process.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-106: Gluing – Output Resources

Name	Description
Component ?	One Component is produced, the input Component with glue applied to it.

6.6.18 HeadBandApplication

Head bands are applied to the hard cover book block.

Table 6-107: HeadBandApplication – Input Resources

Name	Description
Component ?	The prepared book block.
HeadBandApplicationParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-108: HeadBandApplication – Output Resources

Name	Description
Component ?	One Component is produced: the hard cover block with head bands.

6.6.19 HoleMaking

A variety of Machines (e.g., those responsible for stamping and drilling) can perform the **HoleMaking** Process. This postpress Process is needed for different binding techniques (e.g., spiral binding). One or several holes with different shapes can be made that are later on used for binding the book block together.

Table 6-109: HoleMaking – Input Resources

Name	
Component ?	One Component (e.g., a printed Sheet or a pile of Sheets) are modified in the HoleMaking Process.
HoleMakingParams ?	Specific parameters, including hole diameter and positions, used to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-110: HoleMaking – Output Resources

Name	Description
Component ?	A Component with holes (e.g., a book block or a single Sheet) is produced for further postpress Processes.

6.6.20 Inserting

This Process can be performed at several stages in postpress. The Process can be used to describe the labeling of products, labeling of packages or the gluing-in of a **Component** (e.g., a card, Sheet or CD-ROM). Two **Component** Resources are REQUIRED for the **Inserting** Process: the “mother” **Component** and the “child” **Component**. **Inserting** can be a selective Process by means of inserting different “child” **Component** Resources. Information about the placement is needed to perform the Process. Inserting multiple child components is specified as a Combined Process with multiple individual **Inserting** steps.

Table 6-111: Inserting – Input Resources

Name		Description
Component ?		Designates where to insert the child Component .
Component (Child) ?		The Component to be inserted in the mother Component .
InsertingParams ?		Specific parameters (e.g., placement) to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.	

Table 6-112: Inserting – Output Resources

Name	Description
Component	A mother Component is produced containing the inserted child Component .

6.6.21 Jacketing

Jacketing is the Process where the book is wrapped by a jacket that needs to be folded twice. As long as the book is specified and the jacket dimensions are known, there are just a few important details. If the jacketing Device also creases the jacket, this can be described with a Combined Process of **Jacketing** and **Creasing**.

Table 6-113: Jacketing – Input Resources (Sheet 1 of 2)

Name		Description
Component (Book) ?		The book that the jacket is wrapped around.
Component (Jacket) ?		The description of the jacket.

Table 6-113: Jacketing – Input Resources (Sheet 2 of 2)

Name	Description
JacketingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-114: Jacketing – Output Resources

Name	Description
Component ?	The jacketed book.

6.6.22 Labeling

A label can be attached to a bundle. The label can contain information on the addressee, the product, the product quantities, etc., which can be different for each bundle.

Table 6-115: Labeling – Input Resources

Name	Description
Component ?	The Labeling Process labels one Component with a set of labels.
Component (Label) ?	The label to be attached to the Component .
LabelingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-116: Labeling – Output Resources

Name	Description
Component ?	One Component is produced: the labeled Component .

6.6.23 Laminating

In the **Laminating** Process, a plastic film is bonded to one or both sides of a **Component** Resource's media, and adhered under pressure with either a thermal setting or pressure sensitive adhesive.

Table 6-117: Laminating – Input Resources

Name	Description
Component ?	A Component is REQUIRED for Laminating .
LaminatingParams ?	Specific parameters to set up the machinery.
Media (Foil) ?	The laminating foil material.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-118: Laminating – Output Resources

Name	Description
Component ?	One Component is produced: the laminated component.

6.6.24 LooseBinding

LooseBinding describes several types of loose binding processes.

ChannelBinding: Various sizes of metal clamps can be used. The Process can be executed in two ways. In the first, a pile of single Sheets – sometimes together with a front and back cover – is inserted into a U-shaped clamp and

crimped in special machinery. In the second, a pre-assembled cover that includes the open U-shaped clamp is used instead of the U-shaped clamp alone. The thickness of the pile of Sheets determines in both cases the width of the U-shaped clamp to be used for forming the fixed document, which is not meant to be reopened later.

CoilBinding: Another name is *spiral binding*. Metal wire, wire with plastic or pure plastic is used to fasten prepunched Sheets of paper, cardboard or other materials. First, automated machinery forms a spiral of proper diameter and length. The ends of the spiral are then “tucked-in”. Finally, the content is permanently fixed. Note that every time a coil-bound book is opened, a vertical shift occurs as a result of the coil action. This is a characteristic of the Process.

PlasticCombBinding: In this process, a plastic insert wraps through prepunched holes in the substrate. Most often, these holes are rectangular and elongated. After the plastic comb is opened with a special tool, the prepunched block of Sheets – often together with a top and button cover – is inserted onto the “teeth” of the plastic comb. When released from the Machine, the teeth return to their original cylindrical positions with the points tucked into the back-side of the spine area. Special machinery can be used to reopen the plastic comb binding.

RingBinding: In this process, prepunched Sheets are placed in a ring binder. Ring binders have different numbers of rings that are fixed to a metal backbone. In most cases, two, three or four metal rings hold the Sheets together as long as the binding is closed. Depending on the amount of Sheets to be bound together, ring binders of different thickness SHALL be used.

StripBinding: Hard plastic strips are held together by plastic pins, which in turn are bound to the strips with heat. The Sheets to be bound SHALL be prepunched so that the top strip with multiple pins fits through the assembled material. It is then connected to the bottom strip with matching holes for the pins. The binding edge is often compressed in a special Machine before the excess pin length is cut off. The backstrip is permanently fixed with plastic clamping bars and cannot be removed without a special tool.

WireCombBinding: In this process metal wire, wire with plastic or pure plastic is used to fasten prepunched Sheets of paper, cardboard or other such materials. The wire – often formed as a double wire – is inserted into the holes, then curled to create a circular enclosure.

6.6.25 Binding Methods

6.6.25.1 Single-Leaf Binding Methods

Besides the conventional binding methods, there is a multifaceted group of binding methods for single-leaf bindings. This group can again be subdivided into two subtypes: loose-leaf binding and mechanical binding, each of which is described in the sections that follow.

6.6.25.2 Loose-Leaf Binding Method

This binding techniques allow contents to be changed, inserted or removed at will. There are two essential groups of loose-leaf binding systems: those that require the paper to be punched or drilled and those that do not. The RingBinding method, described in the next section, is the most prominent binding in the loose-leaf binding category. Loose-leaf binding methods include:

- RingBinding
- LooseBinding

6.6.25.3 Mechanical Binding Methods

Single leafs are fastened into what is essentially a permanent system that is not meant to be reopened. However, special machinery can be used to reopen some of the mechanical binding systems described below.

In mechanical binding, printing and folding can be done in a conventional manner. The gathered Sheets, however, often require the back to be trimmed, as well as the other three sides. Mechanical bindings are often used for short-run Jobs such as ones that have been printed digitally. The most prominent mechanical binding Processes are described in the sections that follow. Mechanical binding methods include:

- ChannelBinding
- CoilBinding

- PlasticCombBinding
- RingBinding
- StripBinding
- WireCombBinding

Table 6-119: LooseBinding – Input Resources

Name	Description
Component?	The operation requires one component: the block of Sheets to be bound. If Component (Cover) is NOT provided and there is a cover, this Component SHALL be Partitioned, and the first Partition of this Component SHALL specify the Cover
Component (Cover) ?	Covers for LooseBinding .
LooseBindingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-120: LooseBinding – Output Resources

Name	Description
Component?	One Component is produced: the channel-bound component forming an item such as a brochure.

6.6.26 Palletizing

Bundles, stacks, piles or boxes can be loaded onto a pallet.

Table 6-121: Palletizing – Input Resources

Name	Description
Component*	The Palletizing Process describes placing the bundle that is represented by the Component onto a pallet.
Pallet ?	The pallet.
PalletizingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-122: Palletizing – Output Resources

Name	Description
Component?	One Component is produced. It represents the loaded pallet.

6.6.27 Perforating

Perforating describes any Process where a **Component** is perforated.

Table 6-123: Perforating – Input Resources (Sheet 1 of 2)

Name	Description
Component?	This Process consumes one Component : the printed Sheets.
PerforatingParams ?	Details of the Perforating Process.

Table 6-123: Perforating – Input Resources (Sheet 2 of 2)

Name	Description
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-124: Perforating – Output Resources

Name	Description
Component ?	One Component is produced.

6.6.28 ShapeCutting

The **ShapeCutting** Process can be performed using tools such as hollow form punching, perforating or die-cutting equipment.

Table 6-125: ShapeCutting – Input Resources

Name	Description
Component ?	This Process consumes one Component : The Sheets to be cut.
ShapeCuttingParams ?	Details of the ShapeCutting Process.
Tool *	The set of tools (die, counter, blankers, strippers, etc.).
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-126: ShapeCutting – Output Resources

Name	Description
Component*	One or more Component Resources are produced by the ShapeCutting Process.

6.6.29 ShapeDefProduction

This process describes the structural design of a packaging or labels product (e.g., a non rectangular label, a box, a display, a bag, a pouch, etc.). Also, this process typically (but not exclusively) describes the process of designing the shape of a new box using a CAD application. The output of the **ShapeDefProduction** Process can be multiple **ShapeDef** Resources (e.g., when the design of the box results in multiple pieces, such as a box, an object and an insert piece, where the insert piece is fixed to the object to be packed in the box). Another example would be a multi-piece display. The **ShapeDefProduction** Process can be performed by a human operator using a CAD application. In some cases it can be an automated process. Note that **ShapeDefProduction** needs information stored in both **ShapeDefProductionParams** and **ShapeDef** to make a new structural design.

Table 6-127: ShapeDefProduction – Input Resources

Name	Description
RunList ?	A rough drawing or outline (e.g., an EPS) of the ShapeDef that serves as the input for structural design.
ShapeDefProductionParams ?	Parameters for the structural design.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-128: ShapeDefProduction – Output Resources

Name	Description
ShapeDef *	A Resource describing the shape of the product to be produced

6.6.30 Shrinking

The **Shrinking** Process shrinks the shrink-wrap that is wrapped around a bundle. Shrink-wrap foil SHALL be treated in order to shrink.

Note: **Shrinking** does NOT include the wrapping of the **Component** with foil. The actual wrapping is described by the **Wrapping** process. See Section 6.6.55, “Wrapping” on page 433

Table 6-129: Shrinking – Input Resources

Name	Description
Component ?	The bundle including the shrink-wrap media is represented by this Component .
ShrinkingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-130: Shrinking – Output Resources

Name	Description
Component ?	One Component is produced: the bundle including bundle including the shrunk shrink-wrap media.

6.6.31 SpinePreparation

The **SpinePreparation** Process describes the preparation of the spine of book blocks for hard and soft cover book production (e.g., milling and notching).

Table 6-131: SpinePreparation – Input Resources

Name	Description
Component ?	The raw book block.
SpinePreparationParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-132: SpinePreparation – Output Resources

Name	Description
Component ?	The book block with a processed spine.

6.6.32 SpineTaping

SpineTaping describes the Process of applying a tape strip to the spine of a book block. It also describes the Process of applying kraft paper to a hard cover book block.

Table 6-133: SpineTaping – Input Resources (Sheet 1 of 2)

Name	Description
Component ?	The book block that the spine is taped to.
SpineTapingParams ?	Specific parameters to set up the machinery.

Table 6-133: SpineTaping – Input Resources (Sheet 2 of 2)

Name	Description
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-134: SpineTaping – Output Resources

Name	Description
Component ?	The book block with the spine.

6.6.33 Stacking

The **Stacking** Process collects Resource Elements (products) and produces a pile, stack or bundle for delivery. In a standard production each bundle consists of the same amount of identical products, possibly followed by one or more odd-count bundles. In a production with variable data (e.g., newspaper dispatch, demographic production or individual addressed products), each bundle has a variable amount of products, and, in the worst case, each product can be different from the others. The input components are single products; the output components are stacks of this product.

Table 6-135: Stacking – Input Resources

Name	Description
Component ?	The Stacking Process consumes one Component and stacks it onto a stack.
StackingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-136: Stacking – Output Resources

Name	Description
Component ?	One Component is produced: the stack of input Components.

6.6.34 StaticBlocking

The **StaticBlocking** Process puts an electrical charge on a stack in order to hold it together for shipping.

Table 6-137: StaticBlocking – Input Resources

Name	Description
Component ?	The StaticBlocking Process puts an electrical charge on the specified Component .
StaticBlockingParams ?	Specific parameters for the electrical charging.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-138: StaticBlocking – Output Resources

Name	Description
Component ?	The resulting electrically charged Component .

6.6.35 Stitching

Gathered or collected Sheets or Signatures are stitched together with a cover.

Table 6-139: Stitching – Input Resources

Name	Description
Component ?	The only REQUIRED Component is the pile of gathered or collected Sheets, including the cover.
StitchingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-140: Stitching – Output Resources

Name	Description
Component ?	One Component is produced: the gathered or collected Sheets including the cover stitched together.

Example 6-10: Stitching: Combined Process

Components containing staples of different characteristics like shape, width, etc. are defined by a Combined Process.

TBD 2.x Example.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="CombinedStitch"
      JobID="Stitching special" JobPartID="ID123" Type="Combined"
      Types="Stitching Stitching" Status="Ready" Version="1.4">
  <ResourcePool>
    <StitchingParams Class="Parameter" ID="Stitch1" NumberOfStitches="2"
      StapleShape="Butted" Status="Available" StitchPositions="100 700"
      StitchWidth="28.3" WireBrand="Steel" WireGauge="2.3"/>
    <StitchingParams Class="Parameter" ID="Stitch2" NumberOfStitches="2"
      StapleShape="Eyelet" Status="Available" StitchPositions="300 500"
      StitchWidth="42.5" WireBrand="Steel" WireGauge="2.3"/>
    <Component Class="Quantity" ID="Comp1" Status="Available"
      ComponentType="Sheet"/>
    <Component Class="Quantity" ID="Comp2" Status="Unavailable"
      ComponentType="Sheet"/>
  </ResourcePool>
  <ResourceLinkPool>
    <StitchingParamsLink CombinedProcessIndex="0" Usage="Input" rRef="Stitch1"/>
    <StitchingParamsLink CombinedProcessIndex="1" Usage="Input" rRef="Stitch2"/>
    <ComponentLink Usage="Input" rRef="Comp1"/>
    <ComponentLink Usage="Output" rRef="Comp2"/>
  </ResourceLinkPool>
</JDF>
```

6.6.36 Strapping

The Strapping process specifies how straps are wrapped around a bundle. The straps that are used SHOULD be specified as a **MiscConsumable**.

Table 6-141: Strapping – Input Resources (Sheet 1 of 2)

Name	Description
Component ?	The Strapping Process puts straps around a bundle that is represented by a Component .
StrappingParams ?	Specific parameters to set up the machinery.

Table 6-141: Strapping – Input Resources (Sheet 2 of 2)

Name	Description
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-142: Strapping – Output Resources

Name	Description
Component ?	One Component is produced: the strapped Component .

6.6.37 ThreadSealing

Similar to Smythe sewing, **ThreadSealing** involves sewing the Signatures at the spine of the book. After the Signatures are sewn, they are gathered and run through the perfect binder. The perfect binder however does not grind the spine. Instead the binding adhesive (which attaches the cover) envelops the thread that holds the book together. This special thread holds to the glue to create a sewn book with most of the same properties as Smythe sewing.

Table 6-143: ThreadSealing – Input Resources

Name	Description
Component ?	This Process consumes one Component : the printed Sheets.
ThreadSealingParams ?	Details of the ThreadSealing Process.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-144: ThreadSealing – Output Resources

Name	Description
Component ?	One Component is produced.

6.6.38 ThreadSewing

This Process might include a gluing application, which would be used principally between the first and the second Sheet or the last and the last Sheet but one. **Gluing** might also be necessary if different types of paper are used.

Table 6-145: ThreadSewing – Input Resources

Name	Description
Component ?	The operation requires one component: the gathered Sheets.
ThreadSewingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-146: ThreadSewing – Output Resources

Name	Description
Component ?	One Component is produced: the thread-sewn components forming an item such as a raw book block.

6.6.39 Trimming

The **Trimming** Process is performed to adjust a book block or Sheet to its final size. In most cases, it follows a block joining Process, and the Process is often executed as an in-line operation of a production chain. For example,

the binding station might deliver the book blocks to the trimmer. A Combined Process in the trimming machinery would then execute a cut at the front, head and tail in a cycle of two operations. Closed edges of folded Signatures would then be opened while the book block is trimmed to its predetermined dimensions.

The separation of N-up Multiple Products is specified with a **Cutting** Process in front of a **Trimming** Process.

Table 6-147: Trimming – Input Resources

Name	Description
Component ?	The bound book block or Sheet that will be trimmed.
TrimmingParams ?	Specific parameters (e.g., trim size) to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-148: Trimming – Output Resources

Name	Description
Component ?	One Component is produced: the trimmed component.

6.6.40 WebInlineFinishing

The **WebInlineFinishing** Process combines all additional information about inline finishing functionality in connection with Web Printing. In order to describe the **WebInlineFinishing** functionality fully, it is necessary to combine additional Processes like **Stitching**, **Trimming**, **Gluing**, etc.

Table 6-149: WebInlineFinishing – Input Resources

Name	Description
Assembly ?	In context of newspaper printing, Assembly describes how the newspaper Job is sub-divided in physical sections and bound together.
Component ?	Printed webs or ribbons, which will be processed by the WebInlineFinishing Process
ProductionPath ?	ProductionPath describes the paper path that is used through the press and describes exactly one particular product which has to be produced.
Layout ?	Defines how the surfaces of the bindery Signatures of a single Job or Jobs are placed onto the Web(s) or Sheet(s) This information MAY be used for counting the amount of components produced.
WebInlineFinishingParams ?	Additional parameters for production are described by WebInlineFinishingParams
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-150: WebInlineFinishing – Output Resources

Name	Description
Component ?	Describes the finished printed Component out of Web inline finishing equipment. This could be printed and / or folded Sheets or rolls. With one production run, it is possible to produce more than one product / per press run. Component MAY be Partitioned by @ProductPart .

6.6.41 Winding

[New in JDF 1.5](#)

The **Winding** process describes the winding of continuous media or processed components onto a core. The setup is defined in **WindingParams**. The final orientation of the labels on the output roll is specified in **Component/@WindingResult**.

Table 6-151: Winding – Input Resources

Name	Description
Component	Ribbon or Web to be wound.
Media (Core) ?	Core that the input Component is wound around.
WindingParams ?	Setup parameters of the winding process.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-152: Winding – Output Resources

Name	Description
Component ?	The Roll including the core and the wound products. Component/@WindingResult SHALL be evaluated to determine the winding orientation.

6.6.42 Wrapping

Single products, bundles or pallets can be wrapped using bags, bands or wrapping material.

Table 6-153: Wrapping – Input Resources

Name	Description
Component ?	The Wrapping Process wraps a bundle that is represented by a Component .
Component (Wrapper)?	If the wrapping material is preprinted, then Component (Wrapper) represents the wrapping material. Rubber bands and other non-printed material SHOULD be represented as MiscConsumable .
WrappingParams ?	Specific parameters to set up the machinery.
Generic Input Resources*	See Table 6-2 for additional input resources that are valid for all process types.

Table 6-154: Wrapping – Output Resources

Name	Description
Component ?	One Component is produced: the wrapped Component .

6.7 Postpress Processes Structure

6.7.1 Block Production

This subcategory of the postpress Processes merges together all the Processes for making a book block. First the block is compiled using the **Collecting** and **Gathering** Processes. After that, it is combined using one or several of the block joining Processes, including **CoverApplication**, **SpineTaping**, **Stitching** and **ThreadSewing**. The workflow using these Processes eventually produces a **Component** that can be trimmed.

6.7.1.1 Block Compiling

The **Gathering** and **Collecting** Processes are used to position unfolded Sheets and/or folded Sheets in a planned order. These operations set a fixed page sequence in preparation for three-side trimming and binding. Block compiling includes:

- **Collecting**
- **Gathering**
- **Feeding**
- **Winding**

6.7.1.2 Block Joining

The block joining Processes can be grouped into two major subcategories: conventional binding methods, which includes the Processes of **Stitching**, **CoverApplication**, **SpinePreparation**, **SpineTaping**, **ThreadSealing** and **ThreadSewing**; and single-leaf binding methods, which are listed in Section 6.6.30.1, “Single-Leaf Binding Methods”. Together they form a subcategory of block-production Processes. All of these Processes, which are known as block joining Processes, unite Sheets and/or folded Sheets lying loose on top of each other.

There are numerous possible binding methods. The most prominent ones are modeled by the Processes described in the following sections. Many of them can be part of a combined production chain being performed as in-line tasks. Block joining includes:

- **CoverApplication**
- **EndSheetGluing**
- **Gluing**
- **SpinePreparation**
- **SpineTaping**
- **Stitching**
- **ThreadSewing**
-

6.7.2 HoleMaking

- **HoleMaking**

6.7.3 Laminating

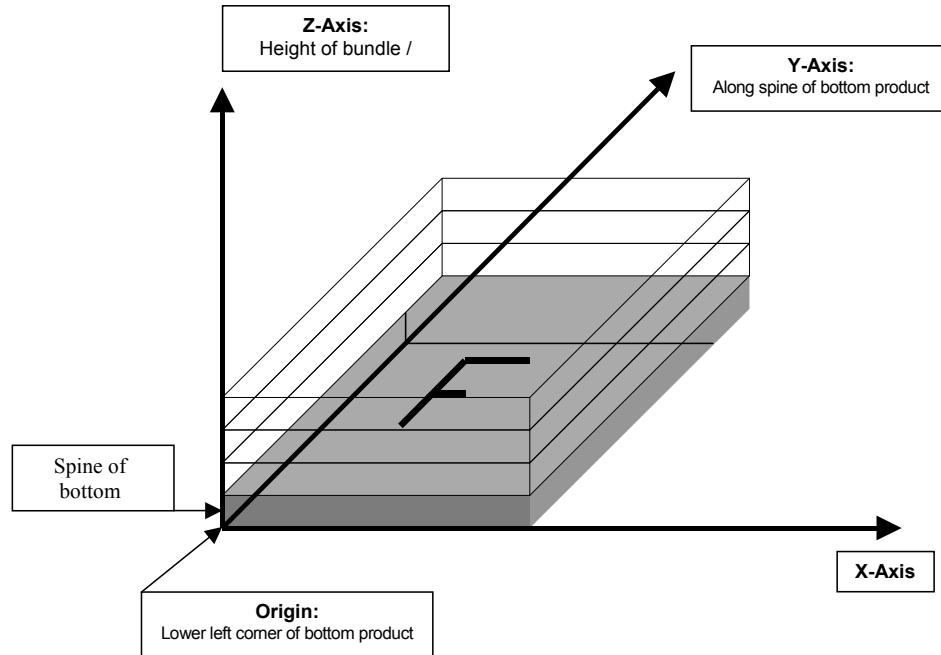
- **Laminating**

6.7.4 Packaging Processes

Packaging Processes include:

- **BoxPacking**
- **Bundling**
- **Labeling**
- **Palletizing**
- **Shrinking**
- **Stacking**
- **Strapping**
- **Wrapping**

Each of these Processes share a common coordinate system as depicted below:

Figure 6-10: Packaging Process Coordinate System

6.7.5 Processes in Hardcover Book Production

The following Processes refer to the production of hard cover books. Several Processes are needed to produce a hard-cover book. Some of them are essential and others are optional. The Processes are:

- CaseMaking:** Production of hard cover book cases.
 - BlockPreparation:** The optional hardcover design elements (e.g., rounding and backing, ribbon band, headband, side gluing and tightbacking) are described in this Process. Application of kraft paper to the book block is described in the **SpineTaping** Process.
 - CasingIn:** In this Process, the case and the prepared book block are brought together.
 - Jacketing:** In the **Jacketing** Process, the jacket is wrapped around the hardcover book.
- Processes in hardcover book production include:
- **BlockPreparation**
 - **CaseMaking**
 - **CasingIn**
 - **Collecting**
 - **Gluing**
 - **HeadBandApplication**
 - **Jacketing**
 - **SpinePreparation**
 - **SpineTaping**
 - **ThreadSealing**
 - **ThreadSewing**

6.7.6 Sheet Processes

Many printing Processes produce Sheets that are processed further in finishing operations. The Web Processes presented in the preceding sections result in Sheets that are treated in much the same way as Sheets produced by Sheet-Fed printing presses. The following Processes describe these Sheet finishing operations. Sheet Processes include:

- **Creasing**
- **Cutting**

- *Embossing*
- *Feeding*
- *Folding*
- *Gathering*
- *Gluing*
- *Palletizing*
- *Perforating*
- *ShapeCutting*
- *ThreadSealing*
- *Winding*

6.7.7 Tip-on/in

The following Processes, **EndSheetGluing**, **Inserting**, are part of the postpress operations. They can be grouped together as the tip-on/in Processes. Both Processes can be performed by hand, tip-on/in Machine or by a press. Tip-on/in includes:

- *EndSheetGluing*
- *Inserting*

6.7.8 Trimming

- *Trimming*.

6.7.9 Web Processes

This subchapter of the postpress Processes is dedicated to Web and ribbon operations (i.e., operations that require a Web or a ribbon to execute). In essence, a ribbon is a Web that has been slit or cross-cut. More specifically, a Web is a continuous strip of **Media** to be used for printing (e.g., paper or foil). This substrate is called “Web” while it is threaded through the printing machinery, but once it has run through the **Cutting** Process and been slit, the Web no longer exists. In its place are ribbons or Sheets.

A ribbon, then, is the part of the Web that enters the folder. If the Web is never slit, however, the Web and the ribbon are identical. Slitting and salvage-trim operations on a Web can result in one or more ribbons. A ribbon can be further subdivided after it has been slit. After the **Cutting** Process, Sheets are treated further. The **Gathering** Process and **Folding** Process also handle Web and ribbon applications.

Chapter 7 Product Intents

Intent Elements are designed to describe a finished product when defining an **XJDF** Job. If an Element or Attribute of an Intent Element is omitted and no additional information is specified in the description, the value defaults to “don’t care”. If an entire Intent Element that specifies a given product feature is omitted, then that feature is not requested. For instance, if a Product Intent description has no **BindingIntent**, then no binding is requested. The characteristics of the product that are not specified through the use of Intent Elements will be selected by the system that Processes the Intent Elements. The system that processes the Product Intent data in an **XJDF** Job ticket MAY insert the details of its selection into the **XJDF** data for the Job. See Section 1.6.2.1, “Conformance Requirements for Support of Attributes and Attribute Values” on page 17 for more information on the handling and processing of systems-specified default values.

All Intent Elements share a set of Subelements that allow a Request for Quote to describe a range of acceptable values for various aspects of the product. These elements, taken together, allow an administrator to provide a specific value for the quote. The section below () describes these Elements.

7.1 Product Description

The products or set of products that are processed during a given workstep MAY be specified in a **ProductList**. This describes a bill of materials. Multiple end products MAY be specified, e.g. when a press sheet of a gang job that contains multiple individual customer jobs is printed.

Product Elements SHOULD contain Intent Elements that describe the details of the desired Product or Product part.

7.1.1 Representation of Product Binding

BindingIntent and **InsertingIntent** SHALL specify how multiple product parts are combined.

7.1.2 Product

The Product Element specifies an individual product or sub product.

A Node should contain at most one **Resource** for each type of Intent Element. If multiple product parts with different intents are needed, each part has its own Product Intent Node.

Table 7-1: Product Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Amount?</i>	integer	Total number of Products or Product parts of this type to produce.
<i>CommentURL?</i>	URL	URL to an external, human-readable description of the Product or Product Part.
<i>DescriptiveName?</i>	string	Human-readable descriptive name of the Product or Product Part.
<i>ExternalID?</i>	NMTOKEN	Identifier of the Product in an MIS.
<i>ID?</i>	ID	Identifier of this product.
<i>IsRoot?</i>	boolean	If true, this Product is a self-contained product. If false, this Product is a child product of another Product , such as a cover or insert. Multiple Product elements MAY be specified, for instance in a gang job.
<i>MaxAmount?</i>	integer	Maximum total number of Products or Product parts of this type to produce including the maximum overage that the Customer is willing to accept.
<i>MinAmount?</i>	integer	Minimum total number of Products or Product parts of this type to produce including the maximum underage that the Customer is willing to accept.

Table 7-1: Product Element (Sheet 2 of 2)

Name	Data Type	Description
<i>ProductType</i> ?	NMTOKEN	Type of product that this component specifies. Values include those from: Table 7-2, “ProductType Attribute Values” on page 440.
<i>ProductTypeDetails</i> ?	string	Additional details of the product: If @ <i>ProductType</i> = “BlankBox” or @ <i>ProductType</i> = “FlatBox”, @ <i>ProductTypeDetails</i> specifies a box type (e.g., [ECMA], [FEFCO] or company internal box type standard).
<i>Comment</i> *	element	Any human-readable text that relates to the product or product part.
<i>GeneralID</i> *	element	Additional identifiers related to the Product or Product Part.
<i>Intent</i> *	element	Container Elements for Intent resources.

— Attribute: ProductType**Table 7-2: ProductType Attribute Values (Sheet 1 of 2)**

Value	Description
<i>BackCover</i>	
<i>BlankBox</i>	Cut, Unfolded box, input for folder-gluer
<i>BlankSheet</i>	A sheet with connected blanks after a die cutting
<i>BlankWeb</i>	A web with connected blanks after a die cutting
<i>Body</i>	Generic content inside of a cover.
<i>Book</i>	
<i>BookBlock</i>	
<i>BookCase</i>	
<i>Box</i>	Convenience packaging that is not envisioned to be protection for shipping.
<i>Brochure</i>	
<i>BusinessCard</i>	
<i>Carton</i>	Protection packaging for shipping.
<i>Cover</i>	
<i>EndSheet</i>	Endsheet for hard cover books.
<i>FlatBox</i>	A folded and glued blank (not opened). Output from a box folder-gluer.
<i>FlatWork</i>	Non-bound, non-folded Products or Products that only have packaging folds.
<i>FrontCover</i>	
<i>Insert</i>	
<i>Jacket</i>	Hard cover case jacket.
<i>Label</i>	
<i>Notebook</i>	A book or block with a set of identical or similar pages, e.g. a writing tablet, where all page fronts have identical content, and all page backs have identical content.
<i>Newspaper</i>	A newspaper-product
<i>Pallet</i>	Loaded pallet of Boxes, Cartons or Component Resources

Table 7-2: ProductType Attribute Values (Sheet 2 of 2)

Value	Description
Poster	
Stack	Stacked Component.

7.1.3 Intent

Intent Elements define the details of products to be produced without defining the Process to produce them. The details of all Product Intents are described in Chapter 7, “Product Intents”.

Table 7-3: Intent Element

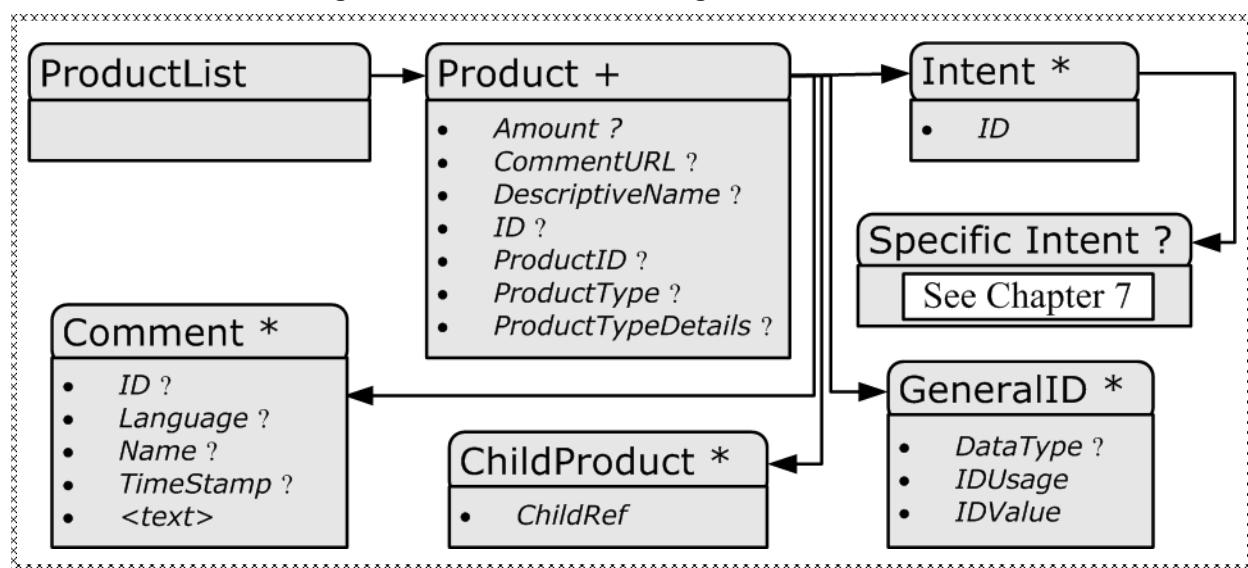
Name	Data Type	Description
Name	enumeration	Type of the Product Intent. A list of valid values is specified in Table 7-4, “Product Intent – Input Resources Intent Elements”.
Product Intent ?	element	Details of the Intent. The node name of the element SHALL be the value of @Name.

7.1.4 Product Intent

A Product Intent is any Intent defined in this chapter (e.g., BindingIntent). There is a separate section for each Product Intent. A Product Intent is a child of a Intent Element

7.1.5 Structure Diagram of ProductList

Figure 7-1 shows the structure of the ProductList. Arrows point to child elements.

Figure 7-1: ProductList– a Diagram of its Structure

7.1.6 Product Intent Descriptions

Product Intent is also described as an XJDF Product element in a ProductList. The following table defines the list of XJDF Intent Elements used to describe Product Intent.

Table 7-4: Product Intent – Intent Elements

Name	Description
BindingIntent ?	This Resource specifies the binding intent for an XJDF Job.
ColorIntent ?	This Resource specifies the type of ink to be used for an XJDF Job.
ContentCheckIntent ?	This Resource specifies the prepress proofing intent for an XJDF Job, using information that identifies the type, quality, brand name and overlay of the proof.
EmbossingIntent ?	This Resource specifies the embossing and/or foil stamping intent for an XJDF Job.
FoldingIntent ?	This Resource specifies the fold intent for an XJDF Job using information that identifies the number of folds, the height and width of the folds, and the folding catalog number.
HoleMakingIntent ?	This Resource specifies the holemaking intent for an XJDF Job.
InsertingIntent ?	This Resource specifies the placing or inserting of one component within another, using information that identifies page location, position and attachment method.
LaminatingIntent ?	This Resource specifies the laminating intent for an XJDF Job using information that identifies whether or not the product is laminated.
LayoutIntent ?	This Resource records the size of the finished pages for the product component.
MediaIntent ?	This Resource describes the media to be used for the product component.
ProductionIntent ?	This Resource specifies the manufacturing intent and considerations for an XJDF Job using information that identifies the desired result or specified manufacturing path.
ShapeCuttingIntent ?	This Resource specifies form and line cutting for an XJDF Job.

7.2 Intent Properties Template

Each of the following sections begins with a brief narrative description of the Product Intent Element. Following that is a list containing details about the properties of the Product Intent Element, as shown below.

After the list describing the Product Intent Properties, each section contains tables that outline the structure of each Product Intent Element and, when applicable, the abstract or Subelement information that pertains to the Product Intent Element structure. The first column contains the name of the Attribute or Element. A template of these tables is also provided below.

Note: for the Intent Properties Template below, the *italicized* text describes the actual text that would be in its place in an actual Product Intent Element definition.

Note also: for the Product Intent Element Structure Template table below: *Cardinality* in the Name column of the Product Intent Element Structure Template table refers to a cardinality symbol, which is either empty or consists of a symbol, such as “?”. Examples described by the Name column include: “**Ink** *” and “**FileSpec** (“*DeviceLinkProfile*”) ?”. For further details, see Section 1.5.4, “Specification of Cardinality”.

Intent Properties Template

Process Resource Pairing: List of Process Resources to which an Intent Element is generally identified with. In practice, the Process Resources will contain the data with which the customer’s intent is fulfilled in production and distribution of the product. This is a list of the primary Resources and not a complete list.

```
<BindingIntent Class="Intent" ID="BI1" Status="Available">
  <BindingType DataType="EnumerationSpan" Actual="Ring"/>
  <RingBinding>
```

```

<HoleType DataType="EnumerationSpan" Range="R4m-DIN-A5 R6m-DIN-A5">
  <Comment Name="R4m-DIN-A5">
    4 equidistant holes on each side of a hexagonal piece of paper
  </Comment>
  <Comment Name="R6m-DIN-A5">
    6 equidistant holes on each side of a hexagonal piece of paper
  </Comment>
</HoleType>
</RingBinding>
</BindingIntent>

```

7.3 AssemblingIntent

This Product Intent element specifies the creation of a composite component by either the assembly, placing or inserting of one component with or within another, using information that identifies page location, position and attachment method. The Containing Product is the receiving Product part. AssemblingIntent MAY be specified multiple times within one product. Products that are used as inserts SHALL have `@ProductType = "Insert"`

Intent Properties

Process Resource Pairing: `InsertingParams, ManualLaborParams`

Table 7-5: AssemblingIntent Intent Element

Name	Data Type	Description
AssemblyItem	element	Each AssemblyItem element describes an individual item that is assembled with the main Product.
BindIn *	element	Each BindIn element describes an individual insert that is glued into this Product.
BlowIn *	element	Each BlowIn element describes an individual insert that is loosely inserted into this Product.
StickOn *	element	Each StickOn element describes an individual child Product that is glued onto this Product. StickOn is typically used for labels.

7.3.1 Element: AssemblyItem

An AssemblyItem element describes any individual item that is assembled with the main Product. Examples of AssemblyItems include stands or frames.

Figure 7-2: Roll-up Display



Table 7-6: Assemble Element

Name	Data Type	Description
<i>ChildRef</i> *	IDREF	Product/@ID of the product that describes the Assemble.

7.3.2 Element: BindIn**Table 7-7: BindIn Element**

Name	Data Type	Description
<i>Folio</i> ?	integer	Index of the parent surface where the insert SHALL be bound in the context of this Product.
<i>Position</i> ?	XYPair	Position of the bottom left corner of the insert in the coordinate system of the surface specified by @Folio after applying all rotations
<i>Orientation</i> ?	Orientation	Orientation of the insert in the coordinate system of the surface specified by @Folio.
<i>ChildRef</i>	IDREF	Product/@ID of the product that describes the insert.
<i>GlueLine</i> ?	element	Details of the Glue used to fasten the insert.

7.3.3 Element: BlowIn**Table 7-8: BlowIn Element**

Name	Data Type	Description
<i>FolioFrom</i> ?	integer	Index of the first valid parent surface where the insert SHALL be placed in the context of this Product.
<i>FolioTo</i> ?	integer	Index of the last valid parent surface where this Product SHALL be placed in the context of the parent Product.
<i>Orientation</i> ?	Orientation	Orientation of the referenced insert in the coordinate system of the surface specified by @Folio
<i>ChildRef</i>	IDREF	Product/@ID of the product that describes the insert.

7.3.4 Element: StickOn**Table 7-9: StickOn Element (Sheet 1 of 2)**

Name	Data Type	Description
<i>Folio</i> ?	integer	Index of the parent surface where the stick-on SHALL be placed in the context of this Product. @Folio SHALL NOT be specified if Location is specified.
<i>Location</i> ?	Location	Location of the stick-on on the Product. See Location for a list of allowed values. Location SHALL NOT be specified if @Folio is specified.
<i>Position</i> ?	XYPair	Position of the bottom left corner of the stick-on in the coordinate system of the surface specified by @Folio and Location after applying all rotations.
<i>Orientation</i> ?	Orientation	Orientation of the stick-on in the coordinate system of the surface specified by @Folio or Location.

Table 7-9: StickOn Element (Sheet 2 of 2)

Name	Data Type	Description
ChildRef	IDREF	Product/@ID of the product that describes the stick-on.
GlueLine?	element	Details of the Glue used to fasten the stick-on.

7.4 BindingIntent

This ResourceProduct Intent element specifies the binding intent for an XJDF Job using information that identifies the desired type of binding and which sides SHALL be bound. All other Products are bound in the order of their appearance in one of the @ChildRefs of the parent Product.

Intent Properties

Process Resource Pairing: BlockPreparationParams, CaseMakingParams, CasingInParams, CoverApplicationParams, EndSheetGluingParams, GluingParams, GlueLine, InsertingParams, JacketingParams, LooseBindingParams, StitchingParams, SpinePreparationParams, SpineTapingParams, StitchingParams, ThreadSealingParams, ThreadSewingParams

Table 7-10: BindingIntent Intent Element (Sheet 1 of 2)

Name	Data Type	Description
BackCoverColor?	NamedColor	Defines the color of the back cover material of the binding.
BackCoverColorDetails?	string	A more specific, specialized or site-defined name for the color. If BackCoverColorDetails is supplied, BackCoverColor SHOULD also be supplied.
BindingColor?	NamedColor	Defines the color of the spine material of the binding.
BindingColorDetails?	string	A more specific, specialized or site-defined name for the color. If BindingColorDetails is supplied, BindingColor SHOULD also be supplied.
BindingOrder?	enumeration	<p>Specifies whether the child Component Resources are to be collected or gathered if multiple child Component Resources are combined.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>None</i> – The Products referenced by Product/@ChildRefs are NOT bound together. Typically used for flatwork Jobs. <i>Collecting</i> – The Products referenced by Product/@ChildRefs are collected on a spine and placed within one another. The first Component is on the outside. <i>Gathering</i> – The child Component Resources are gathered on a pile and placed on top of one another. The first child Product specified by Product/@ChildRefs is on the top.

Table 7-10: BindingIntent Intent Element (Sheet 2 of 2)

Name	Data Type	Description
<i>BindingSide</i> ?	enumeration	Indicates which side are to be bound. Each of these values is intended to identify an edge of the Job. These edges are defined relative to the orientation of the first page in the Job with content on it. Allowed values are: <i>Top</i> <i>Bottom</i> <i>Right</i> <i>Left</i>
<i>BindingType</i> ?	enumeration	Describes the desired binding for the Job. Allowed values are from: Table 7-27, “BindingType Attribute Values”.
<i>ChildRefs</i> ?	IDREFS	Reference to zero or more child products each identified by <i>Product/@ID</i> (e.g., Cover and Body of a Book).
<i>CoverColor</i> ?	NamedColor	Defines the color of the cover material of the binding.
<i>CoverColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If <i>CoverColorDetails</i> is supplied, <i>CoverColor</i> SHOULD also be supplied.
<i>EdgeGluing</i> ?	element	Details of <i>EdgeGluing</i> .
<i>HardCoverBinding</i> ?	element	Details of <i>HardCoverBinding</i> .
<i>LooseBinding</i> ?	element	Details of <i>LooseBinding</i> .
<i>SaddleStitching</i> ?	element	Details of <i>SaddleStitching</i> .
<i>SideStitching</i> ?	element	Details of <i>SideStitching</i> .
<i>SoftCoverBinding</i> ?	element	Details of <i>SoftCoverBinding</i> .
<i>Tabs</i> ?	element	Details of <i>Tabs</i> .
<i>ThreadSewing</i> ?	element	Details of <i>ThreadSewing</i> .

— Attribute: **BindingType**

Table 7-11: BindingType Attribute Values (Sheet 1 of 2)

Value	Description
<i>ChannelBinding</i>	This type of binding can be handled with the <i>LooseBinding</i> Process.
<i>CoilBinding</i>	This type of binding can be handled with the <i>LooseBinding</i> Process.
<i>CornerStitch</i>	Stitch in the corner that is at the clockwise end binding edge. For example, to stitch in the top left corner, set <i>@BindingSide</i> = “ <i>Left</i> ”. This type of binding can be handled with the <i>Stitching</i> Process.
<i>EdgeGluing</i>	Gluing gathered Sheets at one edge of the pile. This Type of Binding can be handled with the <i>Gluing</i> Process. Products of this type are also referred to as padded.
<i>HardCover</i>	This type of binding defines a hard-cover bound book.
<i>LooseBinding</i>	Generic loose Binding - one of “ <i>CoilBinding</i> ”, “ <i>PlasticComb</i> ”, “ <i>Ring</i> ”, “ <i>StripBind</i> ” and “ <i>WireComb</i> ”.

Table 7-11: BindingType Attribute Values (Sheet 2 of 2)

Value	Description
<i>None</i>	This type of binding defines a stack of pages with no additional binding.
<i>PlasticComb</i>	This type of binding can be handled with the LooseBinding Process.
<i>Ring</i>	This type of binding can be handled with the LooseBinding Process.
<i>SaddleStitch</i>	This type of binding can be handled with the Stitching Process.
<i>SideStitch</i>	This type of binding can be handled with the Stitching Process.
<i>SoftCover</i>	This type of binding defines a soft cover bound book. It includes perfect binding.
<i>StripBind</i>	This type of binding can be handled with the LooseBinding Process.
<i>Tape</i>	This type of binding is an inexpensive version of the " <i>SoftCover</i> ".
<i>WireComb</i>	This type of binding can be handled with the LooseBinding Process.

7.4.1 Element: EdgeGluing

Table 7-12: EdgeGluing Element

Name	Data Type	Description
<i>EdgeGlue</i> ?	enumeration	<p>Glue type used to glue the edge of the gathered Sheets.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>ColdGlue</i> <i>Hotmelt</i> <i>PUR</i> – Polyurethane rubber.

7.4.2 Element: HardCoverBinding

Table 7-13: HardCoverBinding Element (Sheet 1 of 3)

Name	Data Type	Description
<i>BlockThreadSewing</i> ?	boolean	Specified if the block is thread sewn.
<i>CoverStyle</i> ?	NMOKEN	<p>Defines the style of the cover board.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Simple</i> – Single layer cover board, see Figure 7-3. <i>Padded</i> – Padded cover board, see Figure 7-4.
<i>EndSheets</i> ?	boolean	<p><i>@EndSheets</i> SHALL be specified if end Sheets SHALL be applied. Additional details of the <i>EndSheets</i> MAY be specified by supplying a Product that is referenced by the parent Product/@ChildRefs with <i>@ProductType = "EndSheet"</i>.</p>
<i>HeadBands</i> ?	boolean	The following case binding choice specifies the use of headbands on a case bound book. If " <i>true</i> ", headbands are inserted both top and bottom.
<i>HeadBandColor</i> ?	NamedColor	Defines the color of the headband.

Table 7-13: HardCoverBinding Element (Sheet 2 of 3)

Name	Data Type	Description
<i>HeadBandColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If HeadBandColorDetails is supplied, HeadBandColor SHOULD also be supplied.
<i>Jacket</i> ?	enumeration	Specifies whether a hard cover jacket is needed and how it is attached. Details of the jacket MAY be described in the <i>Product</i> that is referenced by the parent <i>Product/@ChildRefs</i> whose <i>@ProductType = "Jacket"</i> . Allowed values are: <i>None</i> – No jacket is needed. <i>Loose</i> – The jacket is loosely wrapped. <i>Glue</i> – The jacket is glued to the spine
<i>JacketFoldingWidth</i> ?	double	Dimension of the jacket folds. See JacketingParams for details.
<i>JapanBind</i> ?	boolean	Bind the book block at the open edge, so that the folds are visible on the outside. If not specified, explicitly, this option is never selected.
<i>RegisterRibbon</i> *	element	Number, materials, colors and details of register ribbons.
<i>SpineBrushing</i> ?	boolean	Brushing option for SpinePreparation .
<i>SpineFiberRoughing</i> ?	boolean	Fiber roughing option for SpinePreparation .
<i>SpineGlue</i> ?	enumeration	Glue type used to glue the book block to the cover. Allowed values are: <i>ColdGlue</i> <i>Hotmelt</i> <i>PUR</i> – Polyurethane rubber.
<i>SpineLevelling</i> ?	boolean	Leveling option for SpinePreparation .
<i>SpineMilling</i> ?	boolean	Milling option for SpinePreparation .
<i>SpineNotching</i> ?	boolean	Notching option for SpinePreparation .
<i>SpineSanding</i> ?	boolean	Sanding option for SpinePreparation .
<i>SpineShredding</i> ?	boolean	Shredding option for SpinePreparation .
<i>StripMaterial</i> ?	enumeration	Spine taping strip material. Allowed values are: <i>Calico</i> <i>Cardboard</i> <i>CrepePaper</i> <i>Gauze</i> <i>Paper</i> <i>PaperlinedMules</i> <i>Tape</i>
<i>Thickness</i> ?	double	Specifies thickness of board which is wrapped as front and back covers of a case bound book, in points.

Table 7-13: HardCoverBinding Element (Sheet 3 of 3)

Name	Data Type	Description
<i>TightBacking</i> ?	enumeration	<p>Definition of the geometry of the back of the book block.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Flat</i> – A flat backing. <i>Round</i> – Rounding way. <i>FlatBacked</i> – Backing way. <i>RoundBacked</i> – Rounding way, backing way.

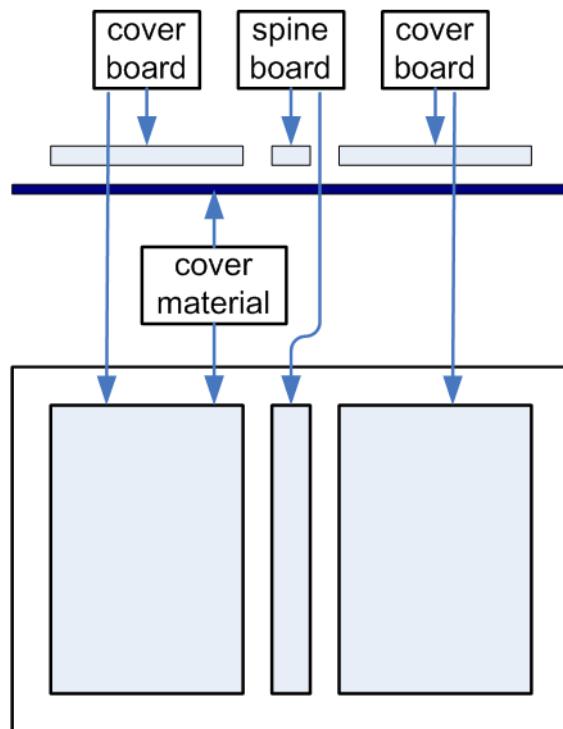
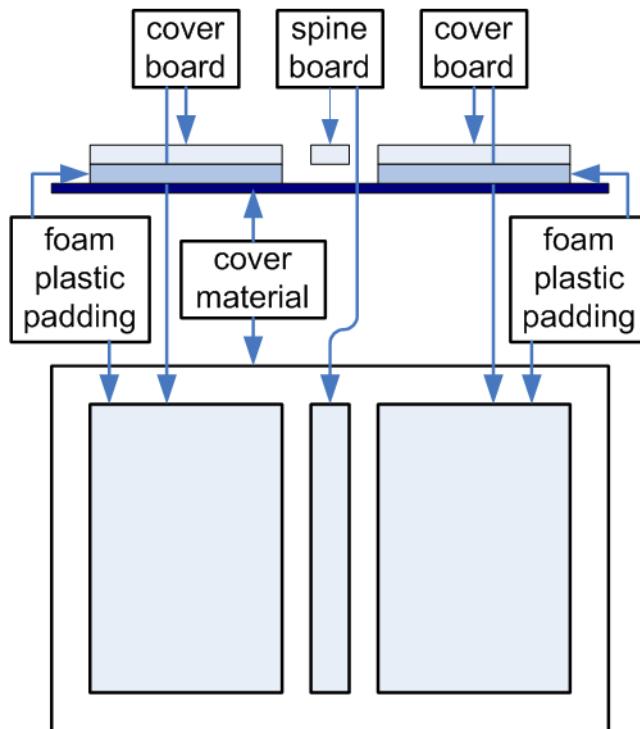
Figure 7-3: Structure of a normal hardcover book

Figure 7-4: Structure of a padded hardcover book

7.4.3 Element: LooseBinding

Table 7-14: LooseBinding Element (Sheet 1 of 3)

Name	Data Type	Description
<i>BinderMaterial?</i>	NMTOKEN	<p>The following describe "<i>RingBinding</i>" binder materials used.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Cardboard</i> – Cardboard with no covering. <i>ClothCovered</i> – Cardboard with cloth covering. <i>Plastic</i> – Binder cover fabricated from solid plastic Sheet material (e.g., PVC Sheet). <i>VinylCovered</i> – Cardboard with colored vinyl covering. <p>Note: <i>@BinderMaterial</i> is for "<i>RingBinding</i>" only.</p>
<i>Brand?</i>	string	Specifies the Binder Brand.

Table 7-14: LooseBinding Element (Sheet 2 of 3)

Name	Data Type	Description
<i>CombShape</i> ?	NMTOKEN	<p>The shape of the comb.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Single</i> – Each “tooth” is made with one wire. <i>SingleCalendar</i> – Each “tooth” is made with one wire and an extension for hanging the bound product is provided in the center. <i>Twin</i> – The shape of each “tooth” is made with a double wire (e.g., <i>Wire-O®</i>). <i>TwinCalendar</i> – The shape of each “tooth” is made with a double wire and an extension for hanging the bound product is provided in the center.
<i>Diameter</i> ?	double	Specifies the diameter of coil, comb or rings, in points.
<i>Material</i> ?	enumeration	<p>The material available for forming the coil binding with "<i>CoilBinding</i>" or wire comb binding with "<i>WireCombBinding</i>".</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Steel</i> – Plain steel. For both binding types. <i>ColorCoatedSteel</i> – Coated steel. For both binding types. <i>Plastic</i> – any kind of plastic. For "<i>CoilBinding</i>" only <p>Note: "<i>Steel-Silver</i>" (for 1.x "<i>WireCombBinding</i>") renamed to "<i>Steel</i>".</p> <p>Note: 2.x @<i>Material</i> replaces 1.x @<i>CoilMaterial</i> and 1.x @<i>WireCombMaterial</i>.</p>
<i>RingShape</i> ?	NMTOKEN	<p>"<i>RingBinding</i>" shapes.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Round</i> <i>Oval</i> <i>D-shape</i> <i>SlantD</i> <p>Note: @<i>RingShape</i> is for "<i>RingBinding</i>" only.</p>
<i>RivetsExposed</i> ?	boolean	<p>The following "<i>RingBinding</i>" choice describes mounting of the ring mechanism in binder case.</p> <p>If "<i>true</i>", the heads of the rivets are visible on the exterior of the binder. If "<i>false</i>", the binder covering material covers the rivet heads.</p> <p>Note: @<i>RivetsExposed</i> is for "<i>RingBinding</i>" only.</p>
<i>ViewBinder</i> ?	NMTOKEN	<p>The values are "<i>RingBinding</i>" clear vinyl outer wrap types and are used on top of a colored base wrap.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Embedded</i> – Printed material is embedded by sealing between the colored and clear vinyl layers during binder manufacturing. <i>Pocket</i> – Binder is designed so that printed material can be inserted between the color and clear vinyl layers after binder manufacturing. <p>Note: @<i>ViewBinder</i> is for "<i>RingBinding</i>" only.</p>

Table 7-14: LooseBinding Element (Sheet 3 of 3)

Name	Data Type	Description
HolePattern ?	element	HolePattern describes the hole pattern that the binder requires. Note that this MAY differ from the holes in the media. For instance the media for a 2 hole ring binder MAY have additional holes that are compatible with a 3 hole ring binder.

7.4.4 Element: SaddleStitching

Table 7-15: SaddleStitching Element

Name	Data Type	Description
StitchNumber ?	integer	Number of stitches used for saddle stitching.

7.4.5 Element: SideStitching

Table 7-16: SideStitching Element

Name	Data Type	Description
StitchNumber ?	integer	Number of stitches used for side stitching.

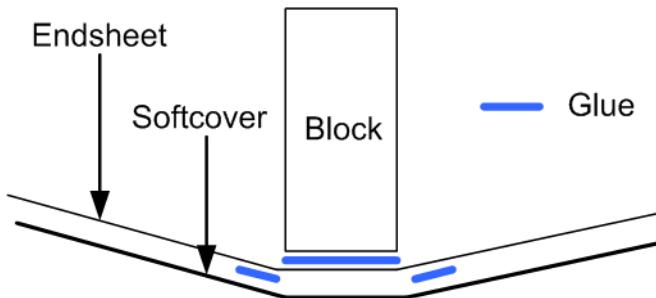
7.4.6 Element: SoftCoverBinding

Table 7-17: SoftCoverBinding Element (Sheet 1 of 2)

Name	Data Type	Description
BlockThreadSewing ?	boolean	Specifies whether the block is also thread sewn.
EndSheets ?	boolean	@EndSheets SHALL be specified if end Sheets SHALL be applied. Additional details of the EndSheets MAY be specified by supplying a Product that is referenced by the parent Product/@ChildRefs whose @ProductType = "EndSheet".
FoldingWidth ?	double	Definition of the dimension of the folding width of the front cover fold. See JacketingParams for details.
FoldingWidthBack ?	double	Definition of the dimension of the folding width of the back cover fold. If not specified, FoldingWidthBack defaults to FoldingWidth.
GlueProcedure ?	enumeration	Glue procedure used to glue the book block to the cover. Allowed values are: <i>Spine</i> <i>SideOnly</i> – Glued at the side or endsheets but not at the spine. " <i>SideOnly</i> " books are also referred to as "layflat" if EndSheets are also specified. See Figure 7-5, "Structure of a book with GlueProcedure = "SideOnly" (Layflat)," on page 472. <i>SingleSide</i> – Swiss Brochure. <i>SideSpine</i> – Both side gluing and spine gluing.

Table 7-17: SoftCoverBinding Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Scoring ?</i>	enumeration	Scoring option for <i>SoftCoverBinding</i> . Values are based on viewing the cover in its flat, prebound state. Allowed values are: <i>TwiceScored</i> <i>QuadScored</i> <i>None</i>
<i>SpineBrushing ?</i>	boolean	Brushing option for <i>SpinePreparation</i> .
<i>SpineFiberRoughing ?</i>	boolean	FiberRoughing option for <i>SpinePreparation</i> .
<i>SpineGlue ?</i>	enumeration	Glue type used to glue the book block to the cover. Allowed values are: <i>ColdGlue</i> <i>Hotmelt</i> <i>PUR – Polyurethane rubber.</i>
<i>SpineLevelling ?</i>	boolean	Leveling option for <i>SpinePreparation</i> .
<i>SpineMilling ?</i>	boolean	Milling option for <i>SpinePreparation</i> .
<i>SpineNotching ?</i>	boolean	Notching option for <i>SpinePreparation</i> .
<i>SpineSanding ?</i>	boolean	Sanding option for <i>SpinePreparation</i> .
<i>SpineShredding ?</i>	boolean	Shredding option for <i>SpinePreparation</i> .

Figure 7-5: Structure of a book with GlueProcedure = "SideOnly" (Layflat)

7.4.7 Element: Tabs

Specifies tabs in a bound document.

Table 7-18: Tabs Element (Sheet 1 of 2)

Name	Data Type	Description
<i>TabBrand ?</i>	string	Strings providing available brand names for the Tabs.
<i>TabCount ?</i>	integer	Number of tabs across all banks. If <i>@TabsPerSet</i> is not an even multiple of <i>@TabsPerBank</i> , the last bank in each set is partially filled.

Table 7-18: Tabs Element (Sheet 2 of 2)

Name	Data Type	Description
<i>TabsPerBank</i> ?	integer	Number of equal-sized tabs in a single bank if all positions were filled. Note that banks can have tabs only in some of the possible positions
<i>TabExtensionDistance</i> ?	double	Distance tab extends beyond the body of the book block, in points.
<i>TabExtensionMylar</i> ?	boolean	If "true", the tab extension will be mylar reinforced.
<i>TabBindMylar</i> ?	boolean	If "true", the tab bind edge will be mylar reinforced.
<i>TabBodyCopy</i> ?	boolean	If "true", Color will be applied not only on tab extension, but also on tab body. Note that lack of body copy allows all tabs within a bank to be printed on a single Sheet.
<i>TabMylarColor</i> ?	NamedColor	Specifies the color of the mylar used to reinforce the tab extension. This is conditional on <i>TabExtensionMylar</i> being "true".
<i>TabMylarColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If <i>TabMylarColorDetails</i> is supplied, <i>TabMylarColor</i> SHOULD also be supplied.

7.4.8 Element: ThreadSewing

Table 7-19: ThreadSewing Element

Name	Data Type	Description
<i>Sealing</i> ?	boolean	If "true", thermo-sealing is needed in <i>ThreadSewing</i> .

7.5 ColorIntent

ColorIntent specifies the color and varnishing of the Product. Each surface SHALL be specified individually in a *SurfaceColor* element.

Intent Properties

Process Resource Pairing: **Color, ColorantControl, ColorCorrectionParams, ColorSpaceConversionParams**

Table 7-20: ColorIntent Intent Element

Name	Data Type	Description
<i>SurfaceColor</i> (Back) ?	element	<i>SurfaceColor</i> [@Surface="Back"] describes the color intent of the back surfaces of the final product.
<i>SurfaceColor</i> (Front) ?	element	<i>SurfaceColor</i> [@Surface="Front"] describes the color intent of the front surfaces of the final product.

7.5.1 Element: SurfaceColor

This element specifies the color configuration of the desired products surface.

Table 7-21: SurfaceColor Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Coatings?</i>	NMTOKENS	<p>Material usually applied to a full surface on press as a protective or gloss-enhancing layer over ink.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Aqueous</i> <i>DullUV</i> New in JDF 1.5 <i>DullVarnish</i> <i>GlossUV</i> New in JDF 1.5 <i>GlossVarnish</i> <i>None</i> – No Coating is requested <i>Protective</i> <i>SatinUV</i> New in JDF 1.5 <i>SatinVarnish</i> <i>Silicone</i> <i>UV</i> – Generic UV <p>Note: Multiple NMTOKENS MAY be selected to indicate multiple coatings.</p> <p>Note: <i>@Surface</i> specifies the surface to which this coating applies.</p> <p>Note: spot coating is specified in <i>@ColorsUsed</i>.</p>
<i>ColorsUsed?</i>	NMTOKENS	<p>Array of colorant separation names that are requested for this <i>SurfaceColor</i>. If specified, <i>@ColorsUsed</i> SHALL contain a list of all separation names used by the Product or a list of spot colors specified by "Spot" which specifies a generic spot color whose details are unknown. "Spot" MAY be specified multiple times in one <i>@ColorsUsed</i> value.</p> <p>Note: If additional information about the colors and colorants is needed, it MAY be specified in the Color ResourceSet. These can be process colors, generic spot colors or named spot colors. In addition, partial (spot) coating is specified by adding NMTOKENS with any value from <i>@Coatings</i>:</p> <p>Note: If additional information about the colors and colorants is needed, it MAY be specified in the Color ResourceSet. These can be process colors, generic spot colors or named spot colors. In addition, partial (spot) coating SHALL be specified by adding NMTOKENS with any value from the list defined in <i>@Coatings</i>:</p>
<i>Coverage?</i>	double	<p>Cumulative colorant coverage percentage. For example, a full Sheet of 100% deep black in CMYK has <i>@Coverage</i> = "400". Typical coverages based on one color plane are:</p> <ul style="list-style-type: none"> <i>Light</i> – 1-9% <i>Medium</i> – 10-35% <i>Heavy</i> – 36+% <p>Note: <i>@Surface</i> specifies the surface to which this coverage applies.</p>

Table 7-21: SurfaceColor Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Surface</i>	enumeration	<p>Allowed values are:</p> <p><i>Back</i> – The back of the product. In case of a wrap-around cover, "<i>Back</i>" specifies the inside of the cover.</p> <p><i>Front</i> – The front of the product. In case of a wrap-around cover, "<i>Front</i>" specifies the outside of the cover.</p>

— Attribute: ColorStandard**Table 7-22: ColorStandard Attribute Values**

Value	Description
<i>CMYK</i>	Generic four color process.
<i>FIRST</i>	Flexographic Image Reproduction Specifications & Tolerances.
<i>GRACOL</i>	General Requirements for Applications in Commercial Offset Lithography
<i>Hexachrome</i>	6 Colors " <i>CYMK</i> " + " <i>Orange</i> " and " <i>Green</i> ".
<i>HIFI</i>	7 Colors " <i>CYMK</i> " + " <i>Red</i> ", " <i>Green</i> " and " <i>Blue</i> ".
<i>JapanColor2001</i>	Japan Color 2001 standard [japancolor].
<i>Monochrome</i>	Generic single color printing condition (e.g., black and white or one single spot color).
<i>SNAP</i>	Specifications for Newsprint Advertising Production
<i>SWOP</i>	Specifications for Web Offset Publications. Registered by ANSI with the ICC as <i>ICC:CGATS TR001</i> pertaining to printing conditions that conform to ANSI CGATS.6 which is based on Publication printing in the US as defined by SWOP, Inc.

7.6 ContentCheckIntent

This Product Intent element specifies the prepress proofing and preflighting intent for an **XJDF Job**.

Intent Properties

Process Resource Pairing: *ApprovalParams, ApprovalDetails, ColorantControl, ColorSpaceConversionParams, ExposedMedia, ImageSetterParams, InterpretingParams, Layout, Media, RenderingParams, ScreeningParams, SeparationControlParams, StrippingParams*

Table 7-23: ContentCheckIntent Intent Element (Sheet 1 of 2)

Name	Data Type	Description
<i>PreflightLevel?</i>	enumeration	<p>Level of content data checking / preflighting. The details are implementation specific.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> • <i>Basic</i> – Check only for severe errors. Examples include missing fonts, unknown file format, incorrect page size, missing passwords. • <i>Extended</i> – Check for additional errors that can degrade output quality and can be resolved by the customer. Examples include: low image resolution, unknown color space details.

Table 7-23: ContentCheckIntent Intent Element (Sheet 2 of 2)

Name	Data Type	Description
ProofItem *	element	Specifies the details of the proofs that are needed. If no ProofItem exists in a ContentCheckIntent , no customer proofs SHALL be provided. Note: ProofItem describes proofs that will be provided to the customer and does not specify internal production proofs.

7.6.1 Element: ProofItem

Table 7-24: ProofItem Element

Name	Data Type	Description
Amount ?	integer	Specifies the total number of copies of this proof that is needed.
ColorType ?	enumeration	Color quality of the proof. Allowed values are: <i>Monochrome</i> – Generic single color printing condition (e.g., black and white or one single spot color). <i>BasicColor</i> – Color does not match precisely. This implies the absence of a color matching system. <i>MatchedColor</i> – Color is matched to the output of the press using a color matching system.
Contract ?	boolean	Requires proof to be a legally binding, accurate representation of the image to be printed (i.e., color quality requirements have been met when the printed piece acceptably matches the proof).
HalfTone ?	boolean	If @HalfTone = "true", the proof SHALL emulate halftone screens.
PageIndex ?	IntegerRangeList	Index of pages that SHALL be proofed in reader order. If @PageIndex is not specified, then all pages SHALL be proofed.
ProofName ?	NMOKEN	Name of the ProofItem. This field SHALL be specified if delivery of a proof is specified in DeliveryParams .
ProofTarget ?	URL	Identifies a remote target for the proof output in a remote proofing environment. This can be either a soft or a hard proofing target. The file to be displayed or output is to be sent to the URL specified in @ProofTarget.

7.7 EmbossingIntent

This Product Intent element specifies the embossing and/or foil stamping intent for an **XJDF** Job using information that identifies whether the product is embossed or stamped, and if desired, the complexity of the affected area.

Intent Properties

Process Resource Pairing: **EmbossingParams**

Table 7-25: EmbossingIntent Intent Element

Name	Data Type	Description
EmbossingItem +	element	Each embossed image is described by one EmbossingItem.

7.7.1 Element: EmbossingItem

Table 7-26: EmbossingItem Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Direction</i> ?	enumeration	<p>The direction of the image.</p> <p>Allowed values are:</p> <p><i>Both</i> – Both debossing and embossing in one stamp.</p> <p><i>Depressed</i> – Debossing.</p> <p><i>Flat</i> – The embossing foil is applied flat. Used for foil stamping.</p> <p><i>Raised</i> – Embossing.</p>
<i>EmbossingTypes</i> ?	NMTOKENS	<p>The strings defined in EmbossingType are whitespace separated combinations of the following tokens.</p> <p>Values include:</p> <p><i>Braille</i> – 6 dot "Braille" embossing.</p> <p><i>BlindEmbossing</i> – Embossed forms that are not inked or foiled. The color of the image is the same as the paper.</p> <p><i>FoilEmbossing</i> – Combines embossing with foil stamping in one single impression.</p> <p><i>FoilStamping</i> – Using a heated die to place a metallic or pigmented image from a coated foil on the paper.</p> <p><i>RegisteredEmbossing</i> – Creates an embossed image that exactly registers to a printed image.</p>
<i>FoilColor</i> ?	NamedColor	Defines the color of the foil material which is used for embossing.
<i>FoilColorDetails</i> ?	string	<p>A more specific, specialized or site-defined name for the color. If FoilColorDetails is supplied, FoilColor SHOULD also be supplied. @FoilColorDetails SHOULD be used to specify specialized foil properties such as holographic or transparent foils. Example combinations of @FoilColor and @FoilColorDetails include:</p> <p>Holographic foils: @FoilColor = "Silver" and @FoilColorDetails = "Holographic";</p> <p>Matte transparent foil: @FoilColor = "white" and @FoilColorDetails = "TransparentMatte".</p>
<i>Height</i> ?	double	The height of the levels. This value specifies the vertical distance between the highest and lowest point of the stamp, regardless of the value of Direction .
<i>ImageSize</i> ?	XYPair	The size of the bounding box of one single image.

Table 7-26: EmbossingItem Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Level</i> ?	enumeration	The level of embossing. Allowed values are: <i>SingleLevel</i> <i>MultiLevel</i> <i>Sculpted</i>
<i>Location</i> ?	enumeration	Position of the embossing on the product. Allowed values are: <i>Back</i> – Back side of a sheet or product <i>Bottom</i> – Bottom of a product <i>Front</i> – Front side of a sheet or product <i>Left</i> – left side of a product, e.g. the spine of a left bound book. <i>Right</i> – Right side of a product, e.g. the spine of a right bound book. <i>Top</i> – Top of a product
<i>Position</i> ?	XYPair	Position of the center of the bounding box of the embossed image in the coordinate system of the surface of the Component that is selected by <i>@Location</i> .
<i>Separation</i> ?	NMOKEN	Separation identifier that specifies a separation that is used for embossing. A value of 0 means that there is no embossing. A value of 1.0 specifies embossing with full depth.
<i>ShapeName</i> ?	string	Name of the embossing tool.

7.8 FoldingIntent

This Product Intent element specifies the straight line folding, creasing and perforating of a product.

Intent Properties

Process Resource Pairing: **CreasingParams**, **CuttingParams**, **Fold**, **FoldingParams**,
 PerforatingParams

Table 7-27: FoldingIntent Intent Element (Sheet 1 of 2)

Name	Data Type	Description
<i>FoldingCatalog</i>	NMOKEN	Describes the folding scheme according to the folding catalog in Figure 8-34, “Fold catalog part 1,” on page 674 and Figure 8-35, “Fold catalog part 2,” on page 675. Value format is: “ <i>Fn-i</i> ” where “n” is the number of finished pages and “i” is either an integer, which identifies a particular fold or the letter “X”, which identifies a generic fold (e.g., “ <i>F6-2</i> ” describes a Z-fold of 6 finished pages, and “ <i>F6-X</i> ” describes a generic fold with 6 finished pages). Note: The folding scheme in this context refers to the folding of the finished product as seen after the cutting, not the folding, of the Sheet as seen in production. See LayoutIntent/@FolioCount for a discussion of pagination of folded end products.

Table 7-27: FoldingIntent Intent Element (Sheet 2 of 2)

Name	Data Type	Description
<i>FoldingDetails</i> ?	NMTOKEN	<i>@FoldingDetails</i> is a system dependent descriptor of the folding. <i>@FoldingDetails</i> MAY be used to differentiate differing fold dimensions with the same general topology, such as asymmetrical Z-folds. <i>@FoldingDetails</i> SHALL NOT be specified if <i>@FoldCatalog</i> is not present
<i>Crease</i> *	element	This describes the details of any creasing operations in the coordinate system of the final product. If no geometrical details are specified in the <i>Crease</i> Element and a <i>@FoldingCatalog</i> is specified, the customer is requesting production creasing.
<i>Fold</i> *	element	This describes the details of folding operations in the sequence described by the value of <i>@FoldingCatalog</i> . <i>Fold</i> SHALL be specified if non-symmetrical folds are requested.
<i>Perforate</i> *	element	Perforate elements describe the details of any perforating operations in the coordinate system of the final product. If no geometrical details are specified in the <i>Perforate</i> Element and a <i>@FoldingCatalog</i> is specified, the customer is requesting production perforation.

7.9 HoleMakingIntent

This Product Intent element specifies the holemaking intent for an **XJDF** Job, using information that identifies the type of holemaking operation or alternatively, an explicit list of holes. This Product Intent element does not specify whether the media will be pre-drilled or the media will be drilled or punched as part of making the product.

Intent Properties

Process Resource Pairing: HolePattern, **HoleMakingParams**, Media

Table 7-28: HoleMakingIntent Intent Element

Name	Data Type	Description
<i>HolePattern</i> *	element	Each <i>HolePattern</i> describes a specific set of holes that SHALL be provided. The coordinate system for applying the holes SHALL be the coordinate system of the product. A <i>HoleMakingIntent</i> with no <i>HolePattern</i> is an explicit request not to provide holes.

7.10 LaminatingIntent

This Product Intent element specifies the laminating intent for an **XJDF** Job using information that identifies whether or not the product is laminated, and if desired, the temperature and thickness of the laminate.

Intent Properties

Process Resource Pairing: **LaminatingParams**

Table 7-29: LaminatingIntent Intent Element

Name	Data Type	Description
<i>Surface</i> ?	enumeration	The surface to be laminated. Allowed values are: <i>Front</i> <i>Back</i> <i>Both</i>
<i>Temperature</i> ?	enumeration	Temperature used in the Laminating Process. Allowed values are: <i>Hot</i> <i>Cold</i>
<i>Texture</i> ?	NMTOKEN	The intended texture of the laminate. Values include: <i>Antique</i> – Rougher than vellum surface. <i>Calendared</i> – Extra-smooth or polished, uncoated paper. <i>Grain</i> <i>Linen</i> – Texture of coarse woven cloth. <i>Matte</i> <i>Smooth</i> <i>Stipple</i> – Fine pebble finish. <i>Vellum</i> – Slightly rough surface.
<i>Thickness</i> ?	double	Thickness of the laminating material. Measured in microns [μm].

7.11 LayoutIntent

This Product Intent element records the size of the finished pages for the product component. It does not, however, specify the size of any intermediate results such as press Sheets. It also describes how the finished pages of the product component are to be imaged onto the finished media. The size definition of the finished media describes the size of a Sheet that is folded to create a product, not the size of a production Sheet (e.g., in the press).

Intent Properties

Process Resource Pairing: **Layout, StrippingParams**

Table 7-30: LayoutIntent Intent Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Bleed</i> ?	double	Bleed of the artwork in points. The value of 0 means NO bleed. A negative value indicates bleed is needed but the value is unknown.

Table 7-30: LayoutIntent Intent Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Dimensions</i> ?	XYPair	Specifies the width (X) and height (Y) in points, respectively, of the trimmed and unfolded (flat) product. For example, <i>Dimensions</i> for a Z-fold is the unfolded dimensions, while <i>@FinishedDimensions</i> is the folded dimensions if known. Use <i>Dimensions</i> if <i>@FinishedDimensions</i> is not known. <i>@Dimensions</i> is provided for the rare case that <i>@FinishedDimensions</i> does not unambiguously define the finished product, due to complex folding schemes. If both values are specified, <i>@FinishedDimensions</i> takes precedence
<i>FinishedDimensions</i> ?	shape	Specifies the width (X), height (Y) and depth (Z) in points, respectively, of the finished product Component after all finishing operations, including folding, trimming, etc. If the Z coordinate is 0, it SHALL be ignored. Only <i>@FinishedDimensions</i> SHOULD be specified if both <i>@FinishedDimensions</i> and <i>@Dimensions</i> are known
<i>NamedDimensions</i> ?	NMTOKEN	Named size (e.g., "A4" or "Letter" that corresponds to the value specified in <i>@FinishedDimensions</i>). If both <i>@NamedDimensions</i> and <i>@FinishedDimensions</i> are specified, then <i>@FinishedDimensions</i> has precedence. See Appendix G, "Media Sizes" on page 1193 for a list of preferred values.
<i>NumberUp</i> ?	XYPair	Specifies a regular, multi-up grid of page cells into which content pages are mapped. The first value specifies the number of columns of page cells and the second value specifies the number of rows of page cells in the multi-up grid (both numbers are integers).
<i>Pages</i> ?	integer	Specifies the number of finished pages (surfaces) of the product component, including blank pages. <i>Pages</i> multiplied with <i>@Dimensions</i> then divided by two (2) identifies the amount of paper that is used in the product. <i>Pages</i> describes the paper usage regardless of document layout. This value SHALL be an even number. For example, the value for <i>Pages</i> for a two-sided booklet with seven Reader Pages would be "8", whether the booklet were saddle stitched or glued.
<i>PrintedPages</i> ?	integer	Number of printed (non-blank) pages. If <i>LayoutIntent/@Sides</i> = "OneSided" or "OneSidedBack", then <i>@PrintedPages</i> = <i>@Pages</i> /2, else <i>@PrintedPages</i> = <i>@Pages</i> .
<i>Sides</i> ?	Sides	<i>Sides</i> SHALL specify which side of the product SHALL be printed.

7.12 MediaIntent

This Product Intent element describes the media to be used for the product component. In some cases, the exact identity of the medium is known, while in other cases, the characteristics are described and a particular stock is matched to those characteristics.

Intent Properties

Process Resource Pairing: **Media**
""

Table 7-31: MediaIntent Intent Element (Sheet 1 of 4)

Name	Data Type	Description
<i>BuyerSupplied</i> ?	boolean	Indicates whether the customer will supply the media. Note that the Media Resource can be used to specify additional media requirements, particularly when the media is supplied by the customer.
<i>Certification</i> ?	string	Name and type of paper certification, e.g. FSC. The details depend on the certification entity.
<i>Flute</i> ?	NMTOKEN	Single, capital letter that specifies the Flute type of corrugated media. Although the classification of flutes using a letter code “A”, “B”, etc., are used very frequently (e.g., in the specification of the order for a box), there seems to be no agreement on the exact numerical specification of those categories. Slightly varying numbers for flute size and frequency can be found between regions (European versus US) and between vendors. Values include those from: Media/@Flute
<i>FluteDirection</i> ?	enumeration	Direction of the fluting. Values are the same as Media/@FluteDirection with slightly different description. Allowed values are: <i>LongEdge</i> – Along the longer axis as defined by LayoutIntent/@Dimensions . <i>ShortEdge</i> – Along the shorter axis as defined by LayoutIntent/@Dimensions . <i>XDirection</i> – Along the X-axis of the LayoutIntent coordinate system <i>YDirection</i> – Along the Y-axis of the LayoutIntent coordinate system
<i>GrainDirection</i> ?	enumeration	Direction of the grain in the coordinate system defined by LayoutIntent/@Dimensions or LayoutIntent/@FinishedDimensions . Allowed values are: <i>Any</i> – No restrictions apply to grain direction. <i>SameDirection</i> – All ordered items SHALL have the same grain direction relative to the finished product. The printer may choose which one. <i>XDirection</i> – Along the X-axis of the LayoutIntent coordinate system <i>YDirection</i> – Along the Y-axis of the LayoutIntent coordinate system.

Table 7-31: MedialIntent Intent Element (Sheet 2 of 4)

Name	Data Type	Description
<i>ISOPaperSubstrate</i> ?	enumeration	<p>The Paper Substrate Type of the Medium from "<i>PS1</i>" through "<i>PS8</i>".</p> <p><i>@ISOPaperSubstrate</i> supersedes <i>@Grade</i> and adds new values to allow for improved papers.</p> <p>If a workflow supports <i>@ISOPaperSubstrate</i>, and both <i>@Grade</i> and <i>@ISOPaperSubstrate</i> are present, it SHALL use <i>@ISOPaperSubstrate</i>.</p> <p><i>@ISOPaperSubstrate</i> type of paper material is defined in accordance with the Print Substrate set forth in [ISO12647-2:2013].</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>PS1</i> – Premium Coated <i>PS2</i> – Improved Coated <i>PS3</i> – Standard Coated Glossy <i>PS4</i> – Standard Coated Matte <i>PS5</i> – Wood-free Uncoated <i>PS6</i> – Super Calendered <i>PS7</i> – Improved Uncoated <i>PS8</i> – Standard Uncoated <p>Note: here in no direct mapping of the 8 values defined in ISO12647-2:2013 to the 5 paper grade values define in ISO12647-2:2004 and earlier. If a legacy implementation requires such mapping, Then the following table MAY be applied.</p>
<i>ISOPaperSubstrateBack</i> ?	enumeration	<p>The Paper Substrate Type of the Medium from "<i>PS1</i>" through "<i>PS8</i>".</p> <p><i>@ISOPaperSubstrateBack</i> supersedes <i>@BackGrade</i> and adds new values to allow for improved papers.</p> <p><i>@ISOPaperSubstrate</i> type of paper material is defined in accordance with the Print Substrate set forth in [ISO12647-2:2013].</p> <p>Allowed values are from: <i>@ISOPaperSubstrate</i>.</p>
<i>MediaColor</i> ?	NamedColor	Color of the media. If more-specific, specialized or site-specific media color names are needed, use <i>MediaColorDetails</i> .
<i>MediaColorDetails</i> ?	string	A more specific, specialized or site-defined name for the media color. If <i>MediaColorDetails</i> is supplied, <i>MediaColor</i> SHOULD also be supplied. Note that there is a one-to-many relationship between entries in <i>MediaColor</i> and <i>MediaColorDetails</i> (e.g., <i>MediaColorDetails</i> values of " <i>Burgundy</i> " and " <i>Ruby</i> " both correspond to a <i>MediaColor</i> of " <i>DarkRed</i> ").

Table 7-31: MedialIntent Intent Element (Sheet 3 of 4)

Name	Data Type	Description
<i>MediaQuality</i> ?	string	<p>Named quality description of the media. For folding carton quality, multiple named quality description systems are in use (e.g., GC1, SBB, etc.). For an overview see http://www.procarton.com/?section=fact_file_publications.</p> <p>When used in a general product description, Media with the same <i>@MediaQuality</i> are identical from the customer point of view. Thus Characteristics such as weight, coatings or recycling percentage are identical whereas lot or sheet dimension may vary based on production or warehousing requirements.</p>
<i>MediaType</i> ?	NMTOKEN	<p>Describes the medium being employed.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>CorrugatedBoard</i> <i>Disc</i> – CD or DVD disc to be printed on. <i>Other</i> – Any other media. For this value <i>MediaTypeDetails</i> SHOULD also be specified <p><i>Paper</i></p> <p><i>SelfAdhesive</i></p> <p><i>Textile</i></p> <p><i>Transparency</i></p> <p><i>Vinyl</i></p>
<i>MediaTypeDetails</i> ?	NMTOKEN	<p>Describes additional details of the medium described in <i>MediaType</i>.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Cloth</i> – Cloth (e.g., for a hard cover book case). <i>Leather</i> – Leather (e.g., for a hard cover book case). <p>Values include those from: <i>Media/@MediaTypeDetails</i></p> <p>Note: values from <i>Media/@MediaTypeDetails</i> are RECOMMENDED. However, some Process related values, such as "<i>DryFilm</i>", SHOULD NOT be used for this Attribute.</p>
<i>NamedWeight</i> ?	string	<p><i>@NamedWeight</i> from Appendix F, "North American and Japanese Media Weight Explained" on page 1189.</p> <p><i>@NamedWeight</i> is for display purposes and SHOULD NOT be specified unless <i>@Weight</i> is also specified.</p>
<i>Opacity</i> ?	enumeration	<p>The opacity of the media. See <i>OpacityLevel</i> to specify the degree of opacity for any of these values.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Opaque</i> – The media is opaque. With two-sided printing the printing on the other side does not show through under normal incident light. <i>Translucent</i> – The media is translucent. For example, translucent media can be used for back lit viewing. <i>Transparent</i> – The media is transparent.
<i>PrePrinted</i> ?	boolean	Indicates whether the media is preprinted.

Table 7-31: MediaIntent Intent Element (Sheet 4 of 4)

Name	Data Type	Description
<i>StockBrand</i> ?	string	Strings providing available brand names. The customer might know exactly what paper is to be used. Example is “Lustro” or “Warren Lustro” even though the manufacturer name is included.
<i>StockType</i> ?	NMTOKEN	Strings describing the available stock. @ <i>StockType</i> defines the base size when calculating North American or Japanese paper weights. See Section F, “North American and Japanese Media Weight Explained” on page 1189. Values include those from: <i>Media</i> /@ <i>StockType</i>
<i>Texture</i> ?	NMTOKEN	The intended texture of the media. Values include those from: <i>Media</i> /@ <i>Texture</i> .
<i>Thickness</i> ?	double	The thickness of the chosen medium. Measured in microns [μm].
<i>Weight</i> ?	double	The intended weight of the media, measured in grammage (g/ m^2) of the media. See Appendix F, “North American and Japanese Media Weight Explained” on page 1189 for an explanation of how to calculate the US weight from the grammage for different stock types.

7.13 ProductionIntent

This Product Intent element specifies the manufacturing intent and considerations for an **XJDF** Job using information that identifies the desired result or specified manufacturing path.

Intent Properties

Process Resource Pairing: All

Table 7-32: ProductionIntent Intent Element (Sheet 1 of 2)

Name	Data Type	Description
<i>PrintPreference</i> ?	enumeration	<p>Intended result or goal.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Balanced</i> – Request for a manufacturing process that balances the requirements for cost, speed and quality. <i>CostEffective</i> – Request for the most cost effective manufacturing process. <i>Fastest</i> – Request for the most time effective manufacturing process. Cost and Quality can be sacrificed for a fast turn-around time. <i>HighestQuality</i> – Request for the manufacturing process which will result in the highest quality.

Table 7-32: ProductionIntent Intent Element (Sheet 2 of 2)

Name	Data Type	Description
<i>PrintProcess</i> ?	NMTOKEN	<p>Print Process requested.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Electrophotography</i> <i>Flexography</i> <i>Gravure</i> <i>Inkjet</i> <i>Lithography</i> – Includes offset printing <i>Letterpress</i> <i>Screen</i> <i>Thermography</i>

7.14 ShapeCuttingIntent

ShapeCuttingIntent describes finishing of products with irregular shapes, including diecutting and adding windows to envelopes.

Intent Properties

Process Resource Pairing: **CuttingParams, ShapeCuttingParams**

Table 7-33: ShapeCuttingIntent Intent Element

Name	Data Type	Description
<i>ShapeCut</i> *	element	Array of all <i>ShapeCut</i> Elements. Used when each shape is exactly specified.

7.14.1 Element: ShapeCut

Table 7-34: ShapeCut Element (Sheet 1 of 2)

Name	Data Type	Description
<i>CutBox</i> ?	rectangle	Specification of a rectangular window. See Section A.2.32, “rectangle” for a definition of the rectangle data type. An orthogonal line MAY be defined by specifying a Rectangle with identical dimensions.
<i>CutDepth</i> ?	enumeration	<p>Allowed values are:</p> <p><i>Full</i> – The form is completely cut out or perforated.</p> <p><i>Partial</i> – The form is not completely cut out or perforated. The exact depth MAY be specified in ShapeCuttingParams.</p>
<i>CutOut</i> ?	boolean	If “ <i>true</i> ”, the inside of a specified shape is to be removed. If “ <i>false</i> ”, the outside of a specified shape is to be removed. An example of an inside shape is a window, while an example of an outside shape is a shaped greeting card.
<i>CutPath</i> ?	PDFPath	Specification of a complex path. This MAY be an open path in the case of a single line.

Table 7-34: ShapeCut Element (Sheet 2 of 2)

Name	Data Type	Description
<i>CutType</i> ?	enumeration	Type of cut or perforation used. Allowed values are: <i>Cut</i> – Full cut. <i>Perforate</i> – Interrupted perforation that does not span the entire Sheet.
<i>MediaRef</i> ?	IDREF	References the material that fills a shape (e.g., an envelope window) that was cut out when <i>@CutOut</i> = "true".
<i>ShapeType</i> ?	enumeration	Describes any precision cutting other than hole making. Allowed values are: <i>Line</i> – The coordinates specified in <i>@CutBox</i> specify the end points of a straight line. <i>Path</i> <i>Rectangular</i> – The coordinates specified in <i>@CutBox</i> specify the lower left and upper right coordinates of a rectangle. <i>Round</i> <i>RoundedRectangle</i> – Rectangle with rounded corners. The coordinates specified in <i>@CutBox</i> specify the outer bounds of the rectangle.

7.15 VariableIntent

This resource specifies the variations of the content for printed data with variable content such as lottery tickets or direct mail.

Table 7-35: VariableIntent Intent Element

Name	Data Type	Description
<i>Area</i> ?	double	Percentage of the document that can contain variable content
<i>AveragePages</i> ?	integer	<i>AveragePages</i> SHALL specify the average number of printed pages in each record.
<i>ChildRefs</i> ?	IDREFS	Product/@ID of the Product elements that describe individual finishing variants. <i>ChildRefs</i> SHALL NOT be specified if InsertingIntent or BindingIntent are specified for this Product.
<i>ColorsUsedFront</i> ?	NMTOKENS	Array of colorant separation names that are required to print the variable part of the documents. The values that are specified in COLORSUSEDFRONT SHALL also be specified in ColorIntent/SurfaceColor/@ColorsUsed . See ColorIntent/SurfaceColor/@ColorsUsed for additional details.
<i>ColorsUsedBack</i> ?	NMTOKENS	Array of colorant separation names that are required to print the variable part of the documents. The values that are specified in ColorsUsedBack SHALL also be specified in ColorIntent/SurfaceColor/@ColorsUsed . See ColorIntent/SurfaceColor/@ColorsUsed for additional details.
<i>MaxPages</i> ?	integer	<i>MaxPages</i> SHALL specify the maximum number of printed pages in each record. <i>MaxPages</i> SHALL NOT be smaller than <i>AveragePages</i> .

Table 7-35: VariableIntent Intent Element

Name	Data Type	Description
MinPages ?	integer	<i>MinPages</i> SHALL specify the minimum number of printed pages in each record. <i>MinPages</i> SHALL NOT be larger than <i>AveragePages</i> .
NumberOfCopies ?	integer	Average number of copies of each record. This value SHALL equal "1" for fully variable data.
VariableType ?	enumeration	Type of variable content. Allowed Values are (in order of rising complexity): <i>OneLine</i> - A single line of text data is variable. OneLine includes simple numbering applications. <i>AddressField</i> - Multiple lines of text data are variable. <i>IdentificationField</i> - The variable data includes a Barcode or QR-Code. <i>Area</i> - The area as defined in Area is fully variable.
VariableQuality ?	enumeration	VariableQuality specifies the desired quality of the variable data. Allowed Values are: <i>Simple</i> - The variable text MAY be recognized as printed by a different technology such as dot matrix or simple inkjet overprints. <i>Imprint</i> - : The variable data SHOULD be similar to the non-variable part but MAY be imprinted. <i>Full</i> - All data SHOULD be printed with the same technology.

Chapter 8 Resources

This chapter provides a list (in alphabetical order) of all Resources.

8.1 Resource

Resource elements represent the “things”, such as inks, plates or glue that are produced or consumed by Processes. **Resource** elements also define the details of Processes, as well as any non-physical computer data such as files used by a Process. They are usually associated with a specific Process. For example, a REQUIRED Input **Resource** of the **DigitalPrinting** Process is the **DigitalPrintingParams Resource**. Most predefined **Resource** elements contain the suffix “Params” in their titles. Examples of **Resource** elements include **FoldingParams** and **ConventionalPrintingParams**.

Table 8-1: Resource Element (Sheet 1 of 3)

Name	Data Type	Description
<i>AgentName?</i>	string	The name of the Agent application that created the Resource . Both the company name and the product name MAY appear, and SHOULD be consistent between versions of the application.
<i>AgentVersion?</i>	string	The version of the Agent application that created the Resource . The format of the version string MAY vary from one application to another, but SHOULD be consistent for an individual application.
<i>Brand?</i>	string	Information, such as the model, part number and/or type, about the Resource being used. Some examples are as follows. <ul style="list-style-type: none">• Premium InkProp Glossy 6x642A• Premium Multipurpose 1234, 88 Bright 24 lb. Bond, 8-1/2 x 11, White Copy Paper Reorder 4711
<i>CommentURL?</i>	URL	URL to an external, human-readable description of the Resource .
<i>ContactRef?</i>	IDREF	If this Element is specified, it describes the owner of the Resource .
<i>DescriptiveName?</i>	string	Human-readable descriptive name of the Resource . It is strongly RECOMMENDED to supply <i>@DescriptiveName</i> in Resource Elements (for example: and ExposedMedia) for communication from applications to humans in order to reference the Resource .
<i>Duration?</i>	duration	Estimated duration during which the Resource will be used.
<i>ExternalID?</i>	NMTOKEN	An ID of the Resource as defined in the MIS system. For instance item codes or article numbers or identifiers on semi-finished products or Resource elements.
<i>GrossWeight?</i>	double	Gross weight of a single Resource , as counted in <i>@Amount</i> , in grams.
<i>ID</i>	ID	Unique identifier of a Resource .
<i>MinLateStatus?</i>	enumeration	Minimum value of NodeInfo/@NodeStatus for the execution of this Node to commence when deadlines are endangered (i.e., when the time defined by NodeInfo/@LastStart or implied by NodeInfo/@LastEnd is approaching). Default value is from: <i>@MinStatus</i> . Allowed values are from: NodeInfo/@NodeStatus .

Table 8-1: Resource Element (Sheet 2 of 3)

Name	Data Type	Description
<i>MinStatus</i> ?	enumeration	Minimum value of <code>NodeInfo/@NodeStatus</code> for the execution of this Node to commence. Allowed values are from: <code>NodeInfo/@NodeStatus</code> .
<i>Orientation</i> ?	Orientation	Named orientation describing the transformation of the orientation of a Resource relative to the ideal Process coordinate that uses this Resource as input or output. If <code>@Orientation</code> is specified for an Output Resource , the Node that processes the Resource is to manipulate the Resource in such a way as to reflect the transformation. The coordinate system of the Resource itself is <i>not</i> modified. At most one of <code>@Orientation</code> or <code>@Transformation</code> SHALL be specified. For details on coordinate systems, see Section 2.5, “Coordinate Systems in JDF XJDF”.
<i>ResourceWeight</i> ?	double	Net weight of a single Resource , as counted in <code>@Amount</code> , in grams.
<i>Start</i> ?	dateTime	Time and date when the usage of the Resource starts.
<i>StartOffset</i> ?	duration	Offset time when the Resource is needed after processing has begun. If both <code>@Start</code> and <code>@StartOffset</code> are specified, <code>@Start</code> has precedence.
<i>Transformation</i> ?	matrix	Matrix describing the transformation of the orientation of a Resource relative to the ideal Process coordinate using this Resource as input or output. If <code>@Transformation</code> is specified for an Output Resource , the Node that processes the Resource is to manipulate the Resource in such a way as to reflect the transformation. The coordinate system of the Resource itself is <i>not</i> modified. At most one of <code>@Orientation</code> or <code>@Transformation</code> SHALL be specified. For details on coordinate systems, see Section 2.5, “Coordinate Systems in JDF XJDF”.
<i>AmountPool</i> ?	element	Definition of partial amounts and pipe parameters for this Resource . The allowed contents of the AmountPool are described in the sections below.
<i>Comment</i> *	element	Any human-readable text that describes the Resource .
<i>GeneralID</i> *	element	Additional identifiers related to the Resource .
<i>IdentificationField</i> *	element	<code>IdentificationField</code> associates bar codes or labels with this Resource .
<i>Location</i> ?	element	Description of details of the location of this Resource . Note: in order to describe multiple locations, Resource MAY be Partitioned by the <code>@Location</code> Partition Key as described in Section 3.13.2.2, “Selecting a Partition”.

Table 8-1: Resource Element (Sheet 3 of 3)

Name	Data Type	Description
Part *	element	The Part Elements identify the parts of a Partitioned Resource that are specified by the Resource element. The structure of the Part Element is defined in Table 3-15, “Part Element” on page 88. For details on Partitioned Resource elements, see Section 3.13.2.2, “Selecting a Partition”. If no Part element is specified, then the Resource applies to the entire ResourceSet. If Multiple Part elements are specified, the Resource describes one physical entity that applies to multiple partitions (e.g., the color plates that apply to all versions of a multi version job)
Specific Resourcer?	element	Details of the Resource . The node name of the element SHALL be the value of ..//ResourceSet/@Name.

8.1.1 Element: Location

Table 8-2: Location Element

Name	Data Type	Description
AddressRef	IDREF	Address of the storage facility. For more information, see Section 10.1, “Address”.
LocationName ?	string	Name of the location (e.g., in MIS). This allows the user to describe distributed Resources. Values include those from: Table C-21, “Input Tray and Output Bin Names” on page 1179. Note: the specified values are for printer locations.
LocID ?	NMTOKEN	Location identifier (e.g., within a warehouse system).

8.1.2 Element: Specific Resourcer

A **Specific Resourcer** is any element defined in a level 2 section of this chapter (e.g., ColorantControl). A **Specific Resourcer** is a child of a **Resource** Element.

8.2 ApprovalParams

This Resource provides the details of an **Approval** Process.

Resource Properties

Resource referenced by:	ProofItem, ConventionalPrintingParams, DigitalPrintingParams
Intent Pairing:	ContentCheckIntent
Input of Processes:	Approval
Output of Processes:	—

Table 8-3: ApprovalParams Resource

Name	Data Type	Description
ApprovalPerson *	element	List of people (e.g., a customer, printer or manager) who can sign the approval.

8.2.1 Element: ApprovalPerson

Table 8-4: ApprovalPerson Element

Name	Data Type	Description
<i>ApprovalRole</i> ?	enumeration	<p>Role of the ApprovalPerson.</p> <p>Allowed values are:</p> <p><i>Approvinator</i> – The decision of this approver immediately overrides the decisions of the other approvers and ends the approval cycle. The "Approvinator" NEED NOT sign for the approval to become valid.</p> <p><i>Informative</i> – The approver is informed of the Approval Process, but the approval is still valid, even without his approval.</p> <p><i>Obligated</i> – The approver SHALL sign the approval.</p>
<i>ApprovalRoleDetails</i> ?	string	Additional details on the @ApprovalRole.
<i>ContactRef</i>	IDREF	Contact (e.g., a customer, printer or manager) who SHALL sign the approval. There SHALL be a Contact [contains (@ContactTypes, "Approver")].

8.3

8.4 ApprovalDetails

The signed **ApprovalDetails** Resource provides the signature that indicates that a Resource has been approved. This is frequently used to model the success of a soft proof, color proof, printing proof or any other sort of proof.

Resource Properties

Resource referenced by: —

Input of Processes: **Any Process**

Output of Processes: **Approval, Verification**

Table 8-5: ApprovalDetails Resource (Sheet 1 of 2) (Sheet 1 of 2)

Name	Data Type	Description
<i>ApprovalState</i> ?	enumeration	<p>Decision made by the approver specified in this ApprovalDetails/ApprovalPerson.</p> <p>Allowed values are:</p> <p><i>Approved</i> – approver approved the Resource.</p> <p><i>ApprovedWithComment</i> – approver approved the Resource but still had some comments.</p> <p><i>Rejected</i> – approver rejected the Resource.</p>
<i>ApprovalStateDetails</i> ?	string	Additional details on the decision made by the approver are specified in this ApprovalDetails/ApprovalPerson . This value provides additional machine readable details of @ApprovalState. Hand written comments and notes MAY be specified in ApprovalDetails/Comment or ApprovalDetails/@CommentURL .

Table 8-5: ApprovalDetails Resource (Sheet 2 of 2) (Sheet 2 of 2)

Name	Data Type	Description
ApprovalPerson ?	element	Details of the person (e.g., a customer, printer or manager) who processed the approval.
Comment ?	element	This Comment provides a container for human readable notes that are provided by the approver.
FileSpec ?	element	The file that contains the approval signature. If FileSpec is not specified, ApprovalDetails is a logical placeholder.

8.5 Assembly

Assembly describes how the sections of one or multiple Jobs or Job Parts are bound together.

Resource Properties

Resource referenced by: Component, CutBlock,

Input of Processes: Collecting, Gathering, SheetOptimizing, Stripping, WebInlineFinishing

Output of Processes: —

Table 8-6: Assembly Resource

Name	Data Type	Description
BinderySignatureIDs ?	NMTOKENS	@BinderySignatureIDs specifies an ordered list of BinderySignature that SHALL be assembled by the method specified in @Order. @BinderySignatureIDs SHALL NOT be present, if @Order = "List".
Order ?	enumeration	<p>Ordering of the individual BinderySignature Elements. Order specifies the topology of the final Assembly.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Collecting</i> – The sections are placed within one another. The first BinderySignature specified in @BinderySignatureIDs is on the outside. An example is a saddle-stitched brochure. See Section 6.6.10, “Collecting” on page 411 <i>Gathering</i> – The sections are placed on top of one another. The first BinderySignature specified in @BinderySignatureIDs is on the top. An example is a perfect bound magazine. See Section 6.6.20, “Gathering” on page 416. <i>None</i> – The sections are not bound. Typically used for flatwork Jobs. <i>List</i> – The Assembly SHALL be fully described with AssemblySection elements.
AssemblySection *	element	Each AssemblySection Elements represents one section that SHALL be gathered. The first AssemblySection SHALL be placed on top, i.e. it is the front of the Assembly . AssemblySection Elements SHALL NOT be specified unless @Order = "List" and SHALL be specified if @Order = "List"..

8.5.1 Element: AssemblySection

An **AssemblySection** represents a recursive set of **BinderySignature** elements. The topology of the **AssemblySection** elements represents the topology of the binding, where sibling **AssemblySection** elements SHALL be gathered from top to bottom and child **AssemblySection** elements SHALL be collected from outside to inside.

Table 8-7: AssemblySection Element

Name	Data Type	Description
<i>BinderySignatureID?</i>	NMTOKEN	@ <i>BinderySignatureID</i> identifies the BinderySignature that this AssemblySection represents. @ <i>BinderySignatureID</i> SHALL be provided if no AssemblySection child elements are provided. @ <i>BinderySignatureID</i> SHALL NOT be provided if AssemblySection child elements are provided.
AssemblySection *	element	Additional child AssemblySection Elements which SHALL be gathered. The resulting set of AssemblySection elements SHALL be collected inside this AssemblySection .

Example 8-1: Perfect Bound (Gathering)

Cover wrapped around a perfect bound (gathering) body

TBD 2.x Example.

```
<Assembly BindingSide="Left" Class="Parameter" ID="ASM000" Order="List"
      Status="Available">
  <AssemblySection AssemblyIDs="Ass_Cover" >
    <AssemblySection AssemblyIDs="Ass_Body1"/>
    <AssemblySection AssemblyIDs="Ass_Body2"/>
    <AssemblySection AssemblyIDs="Ass_Insert"/>
    <AssemblySection AssemblyIDs="Ass_Body3"/>
    <AssemblySection AssemblyIDs="Ass_Body4"/>
  </AssemblySection>
</Assembly>
```

Example 8-2: Saddle-Stitched Brochure (Collecting)

TBD 2.x Example.

```
<Assembly BindingSide="Left" Class="Parameter" ID="ASM000" Order="List"
      Status="Available">
  <AssemblySection AssemblyIDs="Ass_Cover" >
    <AssemblySection AssemblyIDs="Ass_Body1" >
      <AssemblySection AssemblyIDs="Ass_Body2" >
        <AssemblySection AssemblyIDs="Ass_Body3" >
          <AssemblySection AssemblyIDs="Ass_Body4" >
            </AssemblySection>
          </AssemblySection>
        </AssemblySection>
      </AssemblySection>
    </AssemblySection>
  </AssemblySection>
</Assembly>
```

8.6 AssetListCreationParams

This Resource provides controls for the **AssetListCreation** Process.

Resource Properties

Resource referenced by: —

Input of Processes: **AssetListCreation**

Output of Processes: —

Table 8-8: AssetListCreationParams Resource

Name	Data Type	Description
AssetTypes ?	regExp	Specifies what type of assets are to be listed. The regular expression represents the <i>@MimeType</i> of the assets to be listed. The default behavior is to list everything. In case an asset requires a plug-in or extension in order to be opened in an application, this plug-in or extension SHOULD be listed as an asset.
ListPolicy ?	enumeration	Policy that defines which assets SHALL be added to the output RunList . Allowed values are: <i>All</i> – List all referenced assets, including those that are unavailable. <i>Available</i> – List all referenced assets, excluding those that are unavailable.
FileSpec (<i>SearchPath</i>) *	element	An ordered list of search paths that indicates where to search for referenced assets if they are not located in the same directory as the input asset. If no FileSpec is specified, the search path is the directory in which the input asset resides and SHALL NOT be searched recursively.

8.7 BendingParams

BendingParams describes the parameter set for a plate bending and punching Device. A plate is bent and/or punched to fit the press cylinder.

Resource Properties

Resource referenced by: —

Input of Processes: **Bending**

Output of Processes: —

Table 8-9: BendingParams Resource

Name	Data Type	Description
Bend ?	boolean	If "true", indicates that the Device SHALL bend.
Punch ?	boolean	If "true", indicates that the Device SHALL create registration punch holes.
PunchType ?	NMTOKEN	Name of the registration punch scheme (e.g., Bacher).

8.8 BinderySignature

A **BinderySignature** represents both sides of an explicit folding signature with one or more pages. A page typically represents a page in a bound product but MAY also represent a die cut piece of packaging or a single surface of a flat piece such as a postcard.

8.8.1 Pagination for BinderySignatureType Fold

If `@BinderySignatureType = "Fold"`, the distribution of the selected pages on a **BinderySignature** is determined by the 2 attributes: `@FoldCatalog` and `@BinderyOrientation`. The default orientation assumes the Binding Side on the left and the Jog Edge at the bottom.

Resource Properties

Resource referenced by: **StrippingParams**

Input of Processes: —

Output of Processes: —

Table 8-10: BinderySignature Resource (Sheet 1 of 3)

Name	Data Type	Description
<code>BinderySignatureSize ?</code>	XYPair	Size of the BinderySignature including Trim. See Figure 8-1, “BinderySignature Trims,” on page 535 for details.
<code>BinderySignatureType ?</code>	enumeration	<p><code>@BinderySignatureType</code> specifies the type of BinderySignature and the pagination rules.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Fold</i> – The BinderySignature represents a folding dummy. Pagination SHALL be calculated according to Appendix J, “Pagination Catalog” on page 1207. <i>Grid</i> – The BinderySignature represents a grid based layout. The Pagination SHALL be explicitly specified in <code>SignatureCell/@FrontPages</code> and <code>SignatureCell/@BackPages</code>. Grid SHALL be used for flatwork. <i>Die</i> – a layout defined by an existing die.
<code>BindingOrientation ?</code>	Orientation	<p>After folding a BinderySignature, the default coordinate system SHALL be the coordinate system with the binding edge on the left side and the jog edge at the top. <code>@BinderyOrientation</code> defines the transformation that SHALL be applied to the BinderySignature prior to calculating pagination. For Example, a value of <code>"Rotate180"</code> would rotate every page by 180° but retain the pagination whereas a value of <code>"Flip180"</code> would reverse the pagination while retaining the page orientation.</p> <p>For details, see Table 2-4, “Matrices and Orientation values for describing the orientation of a Component” on page 39.</p>
<code>BleedBottom ?</code>	double	<p>Value for the bleed at the bottom side of the BinderySignature.</p> <p>Note: See Section 8.9.2, “On the use of Bleed” on page 534.</p>
<code>BleedLeft ?</code>	double	<p>Value for the bleed at the left side of the BinderySignature.</p> <p>Note: See Section 8.9.2, “On the use of Bleed” on page 534.</p>
<code>BleedRight ?</code>	double	<p>Value for the bleed at the right side of the BinderySignature.</p> <p>Note: See Section 8.9.2, “On the use of Bleed” on page 534.</p>
<code>BleedTop ?</code>	double	<p>Value for the bleed at the top side of the BinderySignature.</p> <p>Note: See Section 8.9.2, “On the use of Bleed” on page 534.</p>

Table 8-10: BinderySignature Resource (Sheet 2 of 3)

Name	Data Type	Description
<i>Bottling</i> ?	enumeration	Bottling SHALL specify the method to use for compensating the bottle angle, which is the slight rotation of a page needed to compensate for the rotation fault introduced when making cross-folds. Allowed values are: <i>All</i> - Compensate all cross-folds <i>Last</i> - Compensate only the bottle angle caused by the final fold <i>None</i> - Do not compensate
<i>DieLayoutRef</i> ?	IDREF	@ <i>DieLayoutRef</i> references a pre-existing die. The content SHALL be imposed to fit the shapes of the referenced DieLayout . DieLayout SHALL NOT be present unless @ <i>BinderySignatureType</i> = "Die".
<i>FoldCatalog</i> ?	FoldCatalog	@ <i>FoldCatalog</i> describes folding of the BinderySignature . Values are defined in the folding catalog in Figure 8-34, "Fold catalog part 1," on page 674 and Figure 8-35, "Fold catalog part 2," on page 675. Constraint: At least one of <i>SignatureCell</i> , @ <i>FoldCatalog</i> or <i>DieLayoutRef</i> SHALL be specified.
<i>NumberUp</i> ?	XYPair	Specifies a regular, multi-up grid of <i>SignatureCell</i> Elements into which content pages are mapped. The first value specifies the number of columns of <i>SignatureCell</i> Elements, and the second value specifies the number of rows of <i>SignatureCell</i> Elements in the multi-up grid (both numbers are integers). When the BinderySignature is Partitioned (e.g., by @ <i>WebName</i>), @ <i>NumberUp</i> MAY be different from leaf to leaf. When the BinderySignature is Partitioned (e.g., by @ <i>WebName</i>), @ <i>NumberUp</i> MAY be different from leaf to leaf.
<i>OutsideGutter</i> ?	boolean	If @ <i>BinderySignatureType</i> is "Grid", this boolean defines whether the outside margins of strip cells have to be taken into account (e.g., if @ <i>OutsideGutter</i> is "false", the Spine (S2) of the strip cells at the left border of the grid is considered to be 0).
<i>StaggerColumns</i> ?	DoubleList	A list of doubles describing the staggering for subsequent columns. The number of entries in the list describes the periodicity of the staggering. Each value gives a factor of the strip cell height ((y value of @ <i>TrimSize</i>) + @ <i>TrimHead</i> + @ <i>TrimFoot</i>) by which to shift the corresponding column (can be negative) (e.g., @ <i>StaggerColumns</i> = "0.0 0.333 0.666" specifies to shift each) <ul style="list-style-type: none"> - 3*n column up by 0% - 3*n+1 column up by 33.3% of the strip cell height - 3*n+2 column up by 66.6% of the strip cell height This Element SHALL NOT be present unless @ <i>BinderySignatureType</i> = "Grid". At most one of @ <i>StaggerColumns</i> or @ <i>StaggerRows</i> SHALL be specified.

Table 8-10: BinderySignature Resource (Sheet 3 of 3)

Name	Data Type	Description
<i>StaggerContinuous</i> ?	boolean	Indicates if the BinderySignature has to be considered as a continuous repetition for staggering. This Attribute SHALL NOT be present unless exactly one of <i>@StaggerRows</i> or <i>@StaggerColumns</i> is specified. Consider a grid with m columns and n rows with <i>@StaggerContinuous</i> = "true". If <i>@StaggerColumns</i> is specified, the BinderySignature SHALL be considered continuous with a height H equal to n multiplied by the strip cell height. If <i>@StaggerColumns</i> has a value of y for a certain column, that column is shifted up (assuming $y > 0$) by an amount equal to y multiplied by the strip cell height (in the same way as described for <i>@StaggerColumns</i>). All content (even partial cells) that falls above H (the top of BinderySignature) is shifted to the bottom such that the top of the shifted content is just below the original bottom cell in the column. For example, if y is 0.666, then the top 66.6% of the top cell is shifted to be just below the original bottom cell. Analogous for <i>@StaggerRows</i> .
<i>StaggerRows</i> ?	DoubleList	A list of doubles describing the staggering for subsequent rows. The number of entries in the list describes the periodicity of the staggering. Each value gives a factor of the strip cell width ((x value of <i>@TrimSize</i>) + <i>@TrimFace</i> + <i>@Spine</i>) by which to shift the corresponding row (can be negative) (e.g., "0.0 0.333 0.666" specifies to shift each) <ul style="list-style-type: none"> – 3*n row right by 0% – 3*n+1 row right by 33.3% of the strip cell width – 3*n+2 row right by 66.6% of the strip cell width This Element SHALL NOT be present unless <i>@BinderySignatureType</i> = "Grid". At most one of <i>@StaggerColumns</i> or <i>@StaggerRows</i> SHALL be specified.
<i>TrimBottom</i> ?	double	Value for cutoff at the bottom side of the BinderySignature . Note: See Section 8.9.3, "On the use of Trim" on page 534.
<i>TrimLeft</i> ?	double	Value for the cutoff at the left side of the BinderySignature . Note: See Section 8.9.3, "On the use of Trim" on page 534.
<i>TrimRight</i> ?	double	Value for the cutoff at the right side of the BinderySignature . Note: See Section 8.9.3, "On the use of Trim" on page 534.
<i>TrimTop</i> ?	double	Value for the cutoff at the top side of the BinderySignature . Note: See Section 8.9.3, "On the use of Trim" on page 534.
<i>SignatureCell</i> *	element	Describes the SignatureCell Elements used in this BinderySignature . SignatureCell Elements are ordered in X-Y direction starting at the lower left-hand corner of the BinderySignature . When both SignatureCell and <i>@FoldCatalog</i> are specified, <i>@FoldCatalog</i> defines the topology of the folding scheme, and the specifics of each individual SignatureCell are described by the SignatureCell Elements. The SignatureCell Elements SHALL have precedence. SignatureCell SHALL NOT be specified if Fold Elements are present.

8.8.2 On the use of Bleed

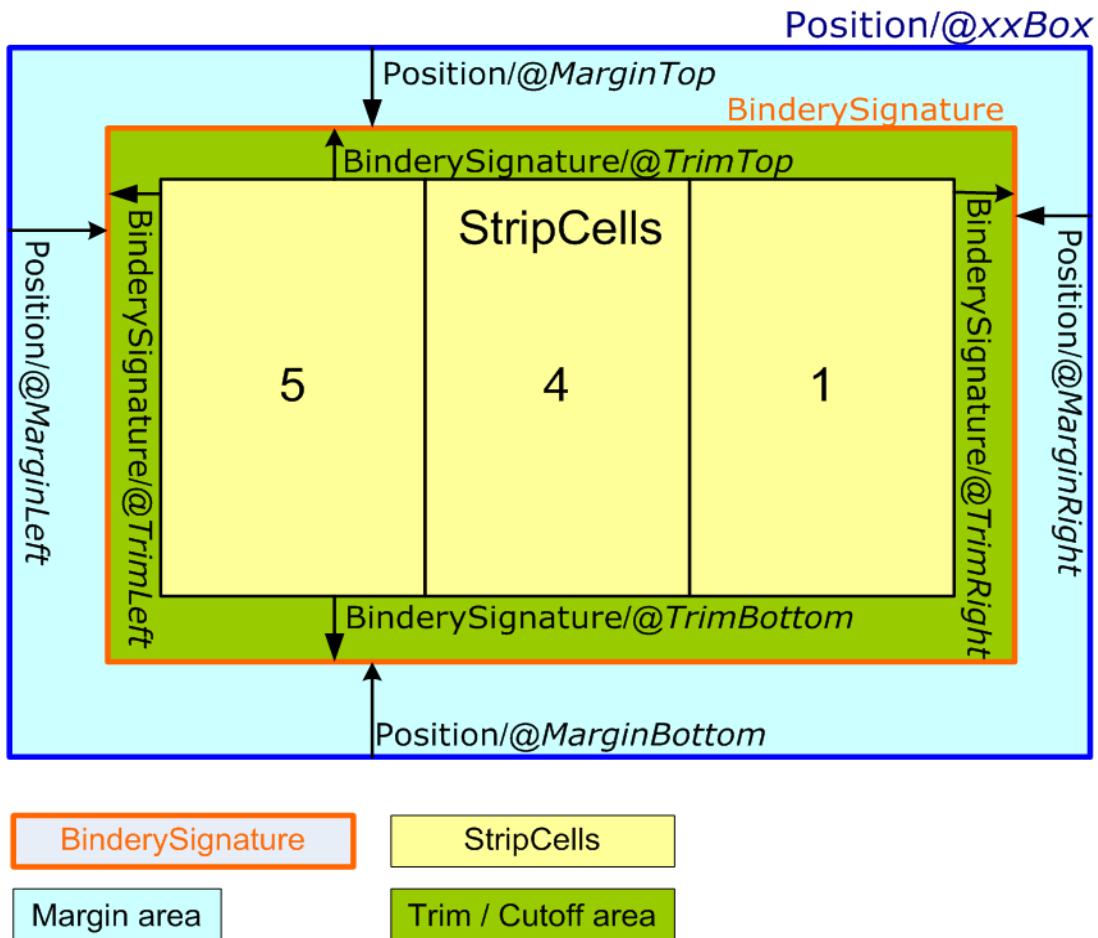
If any Strip Cell belonging to the **BinderySignature** has any bleed value > 0, where a bleed value is **SignatureCell/@BleedFace**, **SignatureCell/@BleedSpine**, **SignatureCell/@BleedHead** or **SignatureCell/@BleedFoot**, then none of the **BinderySignature/@BleedLeft**, **BinderySignature/@BleedRight**, **BinderySignature/@BleedTop** and **BinderySignature/@BleedBottom** SHALL be applied.

If any Strip Cell belonging to the **BinderySignature** has a **SignatureCell/margin** value > 0 (where margin value is: **@Spine**, **@TrimFace**, **@TrimFoot**, **@TrimHead**, **@TrimSize**, **@BackOverflow**, **@FrontOverflow**, **@CutWidthFoot**, **@CutWidthHead** or **@MillingDepth**), then none of **BinderySignature/@BleedLeft**, **BinderySignature/@BleedRight**, **BinderySignature/@BleedTop** and **BinderySignature/@BleedBottom** SHALL be applied.

8.8.3 On the use of Trim

The attributes **@TrimBottom**, **@TrimLeft**, **@TrimRight** and **@TrimTop** are added around the rectangle that is composed of the Strip Cells belonging to the **BinderySignature**. The Strip Cell includes the margins specified by **SignatureCell**. The **Position/@Orientation** is applied to the **BinderySignature/@TrimLeft**, **BinderySignature/@TrimRight**, **BinderySignature/@TrimTop** and **BinderySignature/@TrimBottom** too.

Figure 8-1: BinderySignature Trims



```
maxSectionIndexSeen = 0
maxSectionPages = [0]
for sc in BinderySignature/SignatureCell
    si = sc@SectionIndex
    if (si > maxSectionIndexSeen)
        for index from maxSectionIndexSeen to si - 1:
            maxSectionPages.append(0)
        maxSectionIndexSeen = si
    for page in sc@FrontPages
        maxSectionPages[si] = max(maxSectionPages[si], page)
    for page in sc@BackPages
        maxSectionPages[si] = max(maxSectionPages[si], page)
totalPages = 0
for sectionIndex from 0 to maxSectionIndexSeen
    totalPages += 1 + maxSectionPages[sectionIndex]
return totalPages
```

8.8.4 Element: SignatureCell

SignatureCell Elements describe a set of individual page cells in a **BinderySignature**.

The SignatureCell allow the specification of various distances implicitly defined by the use of a **BinderySignature**. The picture in Figure 8-5 below shows a cell and the different distances inside it leading to the final trim box of the cell in which content will be placed. The size of a strip cell in a Grid is defined by the outermost margin as specified in Figure 8-5.

Note: In practice, SignatureCell values will usually be greater than or equal to zero and have no default.

For more information on spine and trim, see Appendix J, “Pagination Catalog” on page 1207.

The meaning of some attributes in SignatureCell is specified in Appendix J, “Pagination Catalog” on page 1207.

Table 8-11: SignatureCell Element (Sheet 1 of 2)

Name	Data Type	Description
<i>BackOverfold</i> ?	double	(F3) Value for the overfold at the back side.
<i>BackPages</i> ?	IntegerList	Page numbers of the back finished pages of a SignatureCell . The number of entries in <i>@FrontPages</i> and <i>@BackPages</i> SHALL be identical. The entries with an identical index in <i>@FrontPages</i> and <i>@BackPages</i> are back-to-back in the layout. If not specified, the layout is one-sided.
<i>BackSpread</i> ?	IntegerList	Index of SignatureCell Elements that are combined into a spread on the back side.
<i>BleedFace</i> ?	double	(F1) Value for the bleed at the face side.
<i>BleedFoot</i> ?	double	(T1) Value for the bleed at the foot side.
<i>BleedHead</i> ?	double	(H1) Value for the bleed at the head side.
<i>BleedSpine</i> ?	double	(S1) Value for the bleed at the spine side.
<i>CutWidthFoot</i> ?	double	(T3) Amount of paper lost by cutting at the foot side.
<i>CutWidthHead</i> ?	double	(H3) Amount of paper lost by cutting at the head side.
<i>FaceCells</i> ?	IntegerList	List of indices of SignatureCell Elements that form a foldout together with this SignatureCell . The SignatureCell that contains <i>@FaceCells</i> is the parent of the foldout, typically the Page that is attached to the spine. Details of each foldout Page are described by a SignatureCell Element.
<i>FrontOverfold</i> ?	double	(F3) Value for the overfold at the front side.
<i>FrontPages</i> ?	IntegerList	Page numbers of the front finished pages of a SignatureCell . Multiple page cells with the same properties except for the finished pages to which they are assigned MAY be summarized as one SignatureCell with multiple entries in <i>@FrontPages</i> .
<i>FrontSpread</i> ?	IntegerList	Index of SignatureCell Elements that are combined into a spread on the front side.
<i>Mask</i> ?	enumeration	<p>The definition of the clipping mask for the placed graphics.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>None</i> – No mask <i>TrimBox</i> – The mask is derived from the TrimBox as defined by the SignatureCell. <i>BleedBox</i> – The mask is derived from the BleedBox as defined by the SignatureCell. <i>SourceTrimBox</i> – The mask is derived from the TrimBox of the graphical element placed in the SignatureCell. <i>SourceBleedBox</i> – The mask is derived from the BleedBox of the graphical element placed in the SignatureCell. <i>PDL</i> – The mask is derived from the PDL of the graphics. The Attribute <i>@MaskSeparation</i> determines which separation is to be used as the clipping mask for the graphics. <i>DieCut</i> – The mask is the cut line as defined in the DieLayout. <i>DieBleed</i> – The mask is the bleed line as defined in the DieLayout.
<i>MaskBleed</i> ?	double	The distance over which to expand the mask in points.

Table 8-11: SignatureCell Element (Sheet 2 of 2)

Name	Data Type	Description
<i>MaskSeparation</i> ?	NMTOKEN	<i>Color</i> ../Resource/Part/@ <i>Separation</i> of the <i>Color</i> that specifies @ <i>Mask</i> . @ <i>MaskSeparation</i> SHALL be specified if and only if @ <i>Mask</i> = "PDL". <i>Color</i> @ <i>ColorType</i> of this separation SHALL be "DieLine".
<i>MillingDepth</i> ?	double	(S3) Amount of paper cut-off from the spine.
<i>Orientation</i> ?	enumeration	Indicates the orientation of the SignatureCell on the BinderySignature . Allowed values are: <i>Down</i> – 180° rotation. <i>Left</i> – 90° counter-clockwise rotation. <i>Right</i> – 270° counter-clockwise rotation <i>Up</i> – 0° rotation.
<i>Sides</i> ?	Sides	Sides SHALL specify which side of the Media SHALL be printed.
<i>Spine</i> ?	double	(S2) Amount of paper which is not cut-off from the spine. When no Folding is done, this is the left margin. When @ <i>BinderySignatureType</i> = "Grid", the horizontal gutter between cells is @ <i>TrimFace</i> + @ <i>Spine</i> . Note: see Appendix J, "Pagination Catalog" on page 1207.
<i>StationName</i> ?	string	The name of the 1-up station in the die layout. Constraint: if <i>BinderySignature</i> /@ <i>BinderySignatureType</i> = "Die", this Element SHOULD be specified. Constraint: if <i>BinderySignature</i> /@ <i>BinderySignatureType</i> = "Die" and <i>BinderySignature</i> /DieLayout contains more than 1 Station, this Attribute SHALL be specified.
<i>TrimFace</i> ?	double	(F2) Value for the trim distance at the face side. When no Folding is done, this is the right margin. When @ <i>BinderySignatureType</i> = "Grid", the horizontal gutter between cells is @ <i>TrimFace</i> + @ <i>Spine</i> .
<i>TrimFoot</i> ?	double	(T2) Value for the trim distance at the foot side. When no Folding is done, this is the bottom margin. When @ <i>BinderySignatureType</i> = "Grid", the vertical gutter between cells is @ <i>TrimHead</i> + @ <i>TrimFoot</i> .
<i>TrimHead</i> ?	double	(H2) Value for the trim distance at the head side. When no Folding is done, this is the top margin. When @ <i>BinderySignatureType</i> = "Grid", the vertical gutter between cells is @ <i>TrimHead</i> + @ <i>TrimFoot</i> . Note: see Appendix J, "Pagination Catalog" on page 1207.
<i>TrimSize</i> ?	XYPair	Defines the dimensions of the trim box.

```
<!--Stripping Foldout example corresponding to spec example O-24 - with new
```

```
attribute FaceCells-->
```

```
<StrippingParams Class="Parameter" ID="r000005"
```

```
    PartIDKeys="CellIndex" Status="Available">
```

```
    <BinderySignatureRef rRef="r000006"/>
```

```
    <StrippingParams CellIndex="0">
```

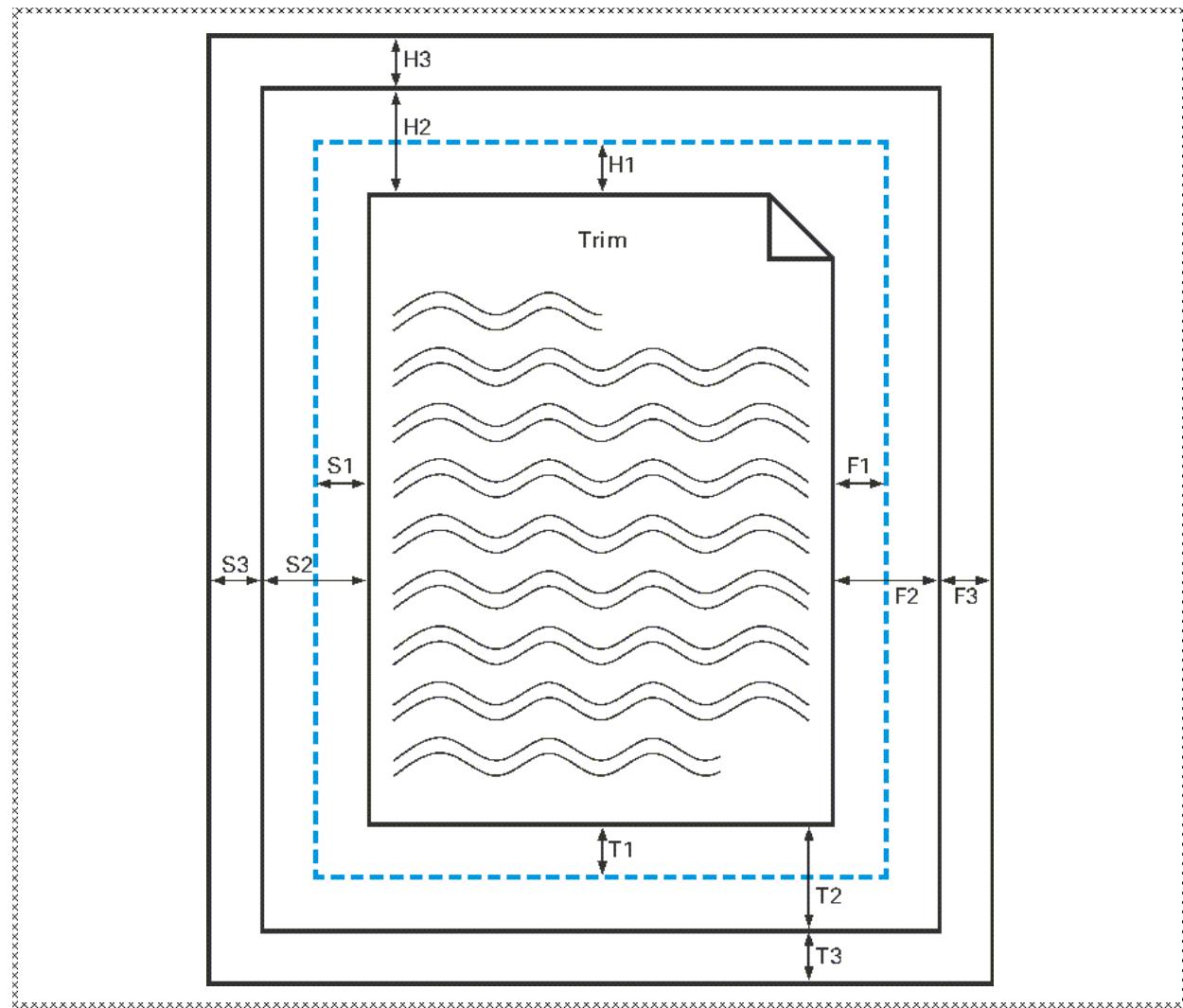
```
        <!--stripcell for the folded out foldout(front page=4)-->
```

```
        <StripCellParams TrimSize="200 400"/>
```

```
</StrippingParams>
<StrippingParams CellIndex="1">
    <!--stripcell for the inner page of the foldout foldout(front page=5)-->
    <StripCellParams TrimSize="300 400"/>
</StrippingParams>
<StrippingParams CellIndex="2">
    <!--stripcell for the inner page of the foldout foldout(front page=0)-->
    <StripCellParams TrimSize="320 400"/>
</StrippingParams>
</StrippingParams>
<BinderySignature Class="Parameter" ID="r000006" Status="Available">
    <!--this is the foldout foldout cell-->
    <SignatureCell BackPages="3" FrontPages="4"/>
    <!--this cell is the inner page of the foldout, i.e. the page that is
        attached to the spine The new attribute FaceCells refers to the cell(s)
        that describe the foldout; in this case the cell to the left. The front
        and back pages of the foldout are listed in the respective cell(s)
    -->
    <SignatureCell BackPages="2" FaceCells="0" FrontPages="5"/>
    <!--this is the cell that has no foldout-->
    <SignatureCell BackPages="1" FrontPages="0"/>
</BinderySignature>
```

8.9 BlockPreparationParams

Figure 8-2: Definition of margins in SignatureCell



This Resource describes the settings of a **BlockPreparation** Process. For the tightbacking there are four different kinds of book forms as shown in Figure 8-6.

Figure 8-3: Tightbacking for Block Preparation

Kinds of Book Forms <code>@TightBacking =</code>	Flat <code>"Flat"</code>	Round <code>"Round"</code>	Flat and Backed <code>"FlatBacked"</code>	Rounded and Backed <code>"RoundBacked"</code>

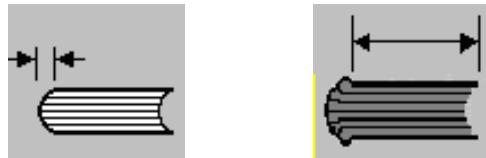
For the rounding and for the backing there are two additional measurement as shown in Figure 8-7.

Figure 8-4: Rounding and Backing for Block Preparation

Measurement	Rounding Way	Backing Way
-------------	--------------	-------------

Figure 8-4: Rounding and Backing for Block Preparation

Attribute $@Rounding = "m"$ $@Backing = "n"$



Resource Properties

Resource referenced by:

Intent Pairing: **BindingIntent**

Input of Processes: **BlockPreparation**

Output of Processes: —

Table 8-12: BlockPreparationParams Resource

Name	Data Type	Description
<i>Backing</i> ?	double	Backing distance in points.
<i>Rounding</i> ?	double	Rounding distance in points.
<i>TightBacking</i> ?	enumeration	Definition of the geometry of the back of the book block. Allowed values are: <i>Flat</i> <i>FlatBacked</i> – Backing way <i>Round</i> – Rounding way <i>RoundBacked</i> – Rounding way, backing way
<i>RegisterRibbon</i> *	element	Description of the register ribbons that are included within the book block.

8.10 BoxFoldingParams

This Resource defines the parameters for folding and gluing blanks to folded flat boxes in a box folder-gluer Device.

Resource Properties

Resource referenced by:

Input of Processes: **BoxFolding**

Output of Processes: —

Table 8-13: BoxFoldingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>BlankDimensionsX</i> ?	DoubleList	X position of folds for an unfolded box beginning from the origin of the coordinate system (left side) increasing from minimum to maximum (expressed in points). See Figure 8-9, “BoxFoldingType Attribute for values of Type00, Type01 and Type02,” on page 548 through Figure 8-12, “BoxFoldingType Attribute for values of Type15 and Type20,” on page 549. The first value of <i>@BlankDimensionsX</i> is the position of the fold marked by X0 in a diagram (e.g., Figure 8-9). The second value of <i>@BlankDimensionsX</i> is the position of the fold marked by X1, and so on. <i>@BlankDimensionsX</i> SHALL NOT be specified unless <i>@BoxFoldingType</i> is also specified.
<i>BlankDimensionsY</i> ?	DoubleList	Y position of folds for of an unfolded box beginning from the origin of the coordinate system (bottom side) increasing from minimum to maximum (expressed in points). See Figure 8-9, “BoxFoldingType Attribute for values of Type00, Type01 and Type02,” on page 548 through Figure 8-12, “BoxFoldingType Attribute for values of Type15 and Type20,” on page 549. The first value of <i>@BlankDimensionsY</i> is the position of the fold marked by Y0 in a diagram (e.g., Figure 8-9). The second value of <i>@BlankDimensionsY</i> is the position of the fold marked by Y2, and so on. <i>@BlankDimensionsY</i> SHALL NOT be specified unless <i>@BoxFoldingType</i> is also present.
<i>BoxFoldingType</i> ?	enumeration	Basic predefined folding types. See the drawings referenced from each defined value below. Each drawing is shown from the print side with the lid at the top. Each type is described with a sequence of <i>BoxFoldAction</i> Elements. The most common sequences (folding types) are predefined, All other are 'special' and SHALL be described in detail. Allowed values are: Type00 – Special type for boxes that are not pre-defined. See Figure 8-9. Type01 – see Figure 8-9. Type02 – see Figure 8-9. Type03 – see Figure 8-10. Type04 – see Figure 8-10. Type10 – see Figure 8-10. Type11 – see Figure 8-11. Type12 – see Figure 8-11. Type13 – see Figure 8-11. Type15 – see Figure 8-12. Type20 – see Figure 8-12.
<i>BoxFoldAction</i> *	element	Individual work step in a Box folder-gluer. The sequence of <i>BoxFoldAction</i> and <i>GlueLine</i> Elements defines the sequence of work steps. The first Element is applied first.

Table 8-13: BoxFoldingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
GlueLine *	element	Specification of a glue line. The GlueLine is applied to the blank in the coordinate system of the folder gluer at the state after all prior BoxFoldAction Elements have been applied. The sequence of BoxFoldAction and GlueLine Elements defines the sequence of work steps. The first Element is applied first.

8.10.1 Element: BoxFoldAction

BoxFoldAction describes an action in the folder-gluer that is perpendicular or diagonal to the movement path of the blank.

Table 8-14: BoxFoldAction Element

Name	Data Type	Description
<i>FoldIndex</i>	XYPair	Identification of the upper right corner of the flap or fold that is affected by this BoxFoldAction. The first value of the XYPair refers to an indexed fold in <i>@BlankDimensionsX</i> ; the second value of the XYPair refers to an indexed fold in <i>@BlankDimensionsY</i> . If either X or Y spans multiple flaps, it SHALL be set to -1.
<i>Action</i>	enumeration	Individual Action in the folder gluer. Allowed values are from: Table 8-17, “Action Attribute Values” on page 546.

— Attribute: Action

Table 8-15: Action Attribute Values (Sheet 1 of 2)

Value	Description
<i>LongFoldLeftToRight</i>	For a drawing, see Figure 8-8, “Folding examples for some values of BoxFoldAction/@Action,” on page 547.
<i>LongFoldRightToLeft</i>	
<i>LongPreFoldLeftToRight</i>	
<i>LongPreFoldRightToLeft</i>	For a drawing, see Figure 8-8, “Folding examples for some values of BoxFoldAction/@Action,” on page 547.
<i>FrontFoldComplete</i>	For a drawing, see Figure 8-8, “Folding examples for some values of BoxFoldAction/@Action,” on page 547.
<i>FrontFoldDiagonal</i>	
<i>FrontFoldCompleteDiagonal</i>	For a drawing, see Figure 8-8, “Folding examples for some values of BoxFoldAction/@Action,” on page 547.
<i>BackFoldComplete</i>	For a drawing, see Figure 8-8, “Folding examples for some values of BoxFoldAction/@Action,” on page 547.
<i>BackFoldDiagonal</i>	
<i>BackFoldCompleteDiagonal</i>	
<i>ReverseFold</i>	A “ReverseFold” is topologically equivalent to “FrontFoldDiagonal” but uses different equipment with other restrictions on Media weight and size and is therefore specified individually. For a drawing, see Figure 8-8, “Folding examples for some values of BoxFoldAction/@Action,” on page 547.

Table 8-15: Action Attribute Values (Sheet 2 of 2)

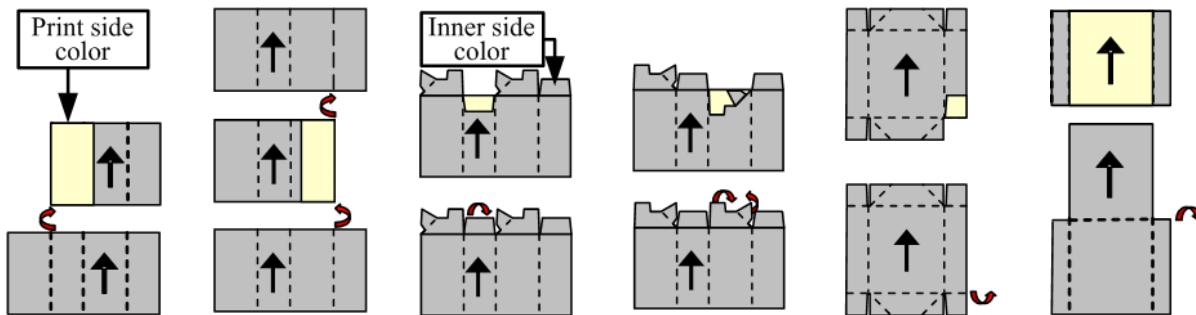
Value	Description
<i>Milling</i>	
<i>Rotate90</i>	90° counter-clockwise rotation
<i>Rotate180</i>	180° rotation
<i>Rotate270</i>	90° clockwise rotation

Example 8-3: BoxFoldingParams/BoxFoldAction

For instance, processing a Type01 blank (Figure 8-9, “BoxFoldingType Attribute for values of Type00, Type01 and Type02,” on page 548) has the following actions:

TBD 2.x Example.

```
<BoxFoldingParams Class="Parameter" ID="BFP000" Status="Available">
  <BoxFoldAction FoldIndex="0 -1" Action="LongPreFoldLeftToRight"/>
  <BoxFoldAction FoldIndex="2 -1" Action="LongPreFoldRightToLeft"/>
  <BoxFoldAction FoldIndex="1 -1" Action="LongFoldLeftToRight"/>
  <BoxFoldAction FoldIndex="3 -1" Action="LongFoldRightToLeft"/>
</BoxFoldingParams>
```

Figure 8-5: Folding examples for some values of BoxFoldAction/@Action

*LongFoldLeft LongPreFoldR FrontFoldCom FrontFoldCom- BackFoldComp ReverseFold
ToRight ightToLeft pletepleteDiagonal lete*

Dimensions and Actions for below Figures:

- Shown from print side, lid at the top, Arrow is transport direction in folder-gluer.
- In the folder-gluer the blank box is fed with the print side down.
- From this point of view all folds are made toward the -z axis.
- For front and back folds, pay attention to transport direction

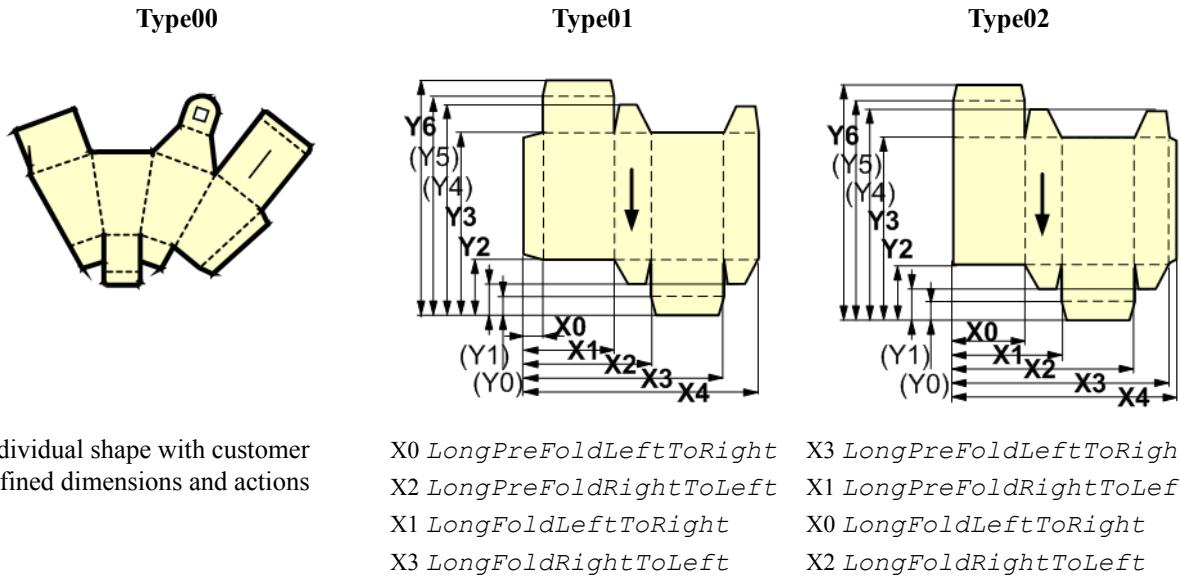
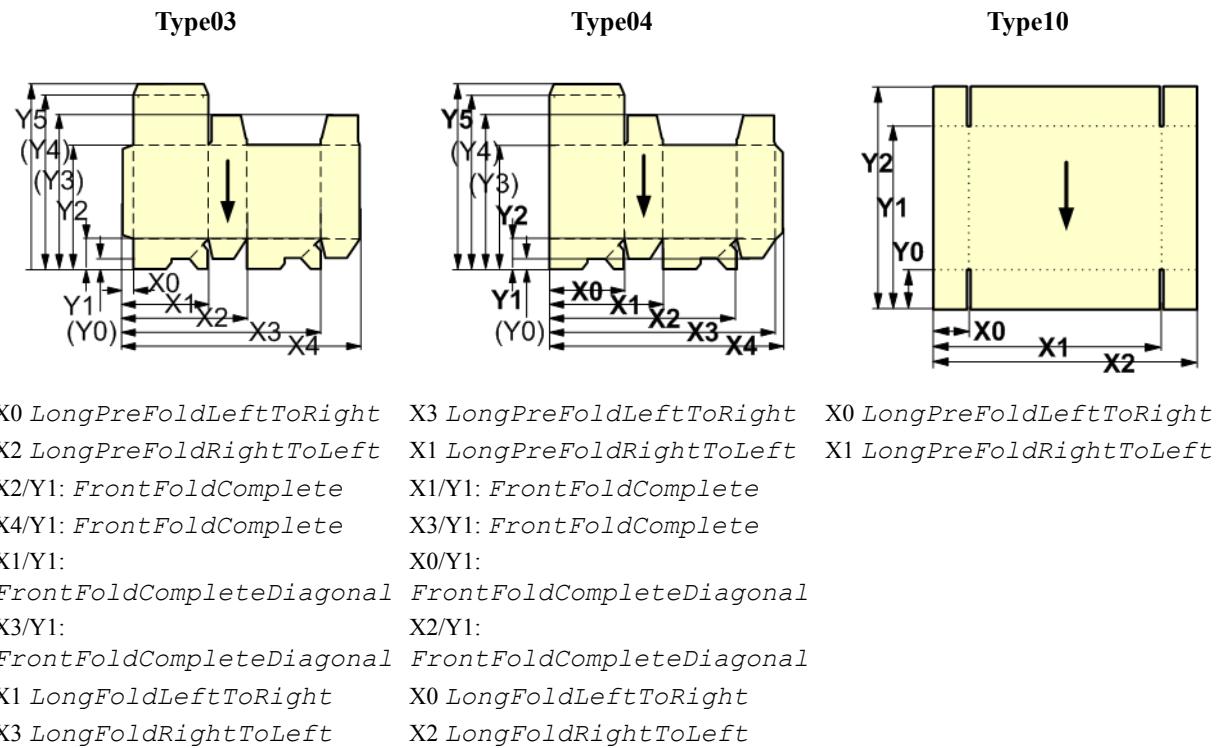
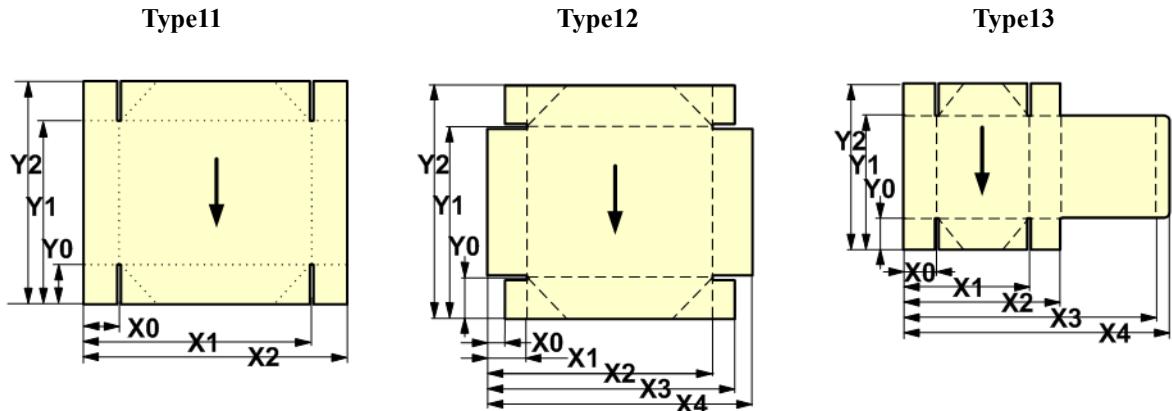
Figure 8-6: BoxFoldingType Attribute for values of Type00, Type01 and Type02**Figure 8-7: BoxFoldingType Attribute for values of Type03, Type04 and Type10**

Figure 8-8: BoxFoldingType Attribute for values of Type 11, Type12 and Type13



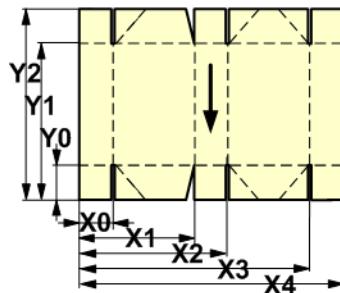
X0/Y0: *FrontFoldComplete*
 X2/Y0: *FrontFoldComplete*
 X0/Y2: *BackFoldComplete*
 X2/Y2: *BackFoldComplete*
 X1/Y0:
FrontFoldCompleteDiagonal
 X1/Y2:
BackFoldCompleteDiagonal
 X0 *LongFoldLeftToRight*
 X2 *LongFoldRightToLeft*

X1/Y0:
FrontFoldCompleteDiagonal
 X1/Y2:
BackFoldCompleteDiagonal
 X0 *LongFoldLeftToRight*
 X2 *LongFoldRightToLeft*

X0/Y0: *FrontFoldComplete*
 X2/Y0: *FrontFoldComplete*
 X0/Y2: *BackFoldComplete*
 X2/Y2: *BackFoldComplete*
 X1/Y0:
FrontFoldCompleteDiagonal
 X1/Y2:
BackFoldCompleteDiagonal
 X0 *LongFoldLeftToRight*
 X2 *LongFoldRightToLeft*

Figure 8-9: BoxFoldingType Attribute for values of Type15 and Type20

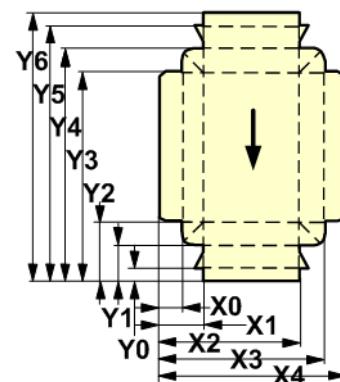
Type15



X0/Y0: *FrontFoldComplete*
 X2/Y0 *FrontFoldComplete*
 X4/Y0 *FrontFoldComplete*
 X0/Y2 *BackFoldComplete*
 X2/Y2 *BackFoldComplete*
 X4/Y2 *BackFoldComplete*
 X1/Y0
FrontFoldCompleteDiagonal

(continued from previous column)
 X3/Y0 *FrontFoldCompleteDiagonal*
 X1/Y2 *BackFoldCompleteDiagonal*
 X3/Y2 *BackFoldCompleteDiagonal*
 X0 *LongFoldLeftToRight*
 X3 *LongFoldRightToLeft*
 X2 *LongFoldRightToLeft*

Type20



X0 *LongFoldLeftToRight*
 X3 *LongFoldRightToLeft*

8.11 BoxPackingParams

This Resource defines the parameters for packing a box of components. Details of the box used for **BoxPacking** can be found in the **Component (Box)** Resource that is also an input of the **BoxPacking** Process.

Resource Properties

Resource referenced by:

Intent Pairing:

Input of Processes: **BoxPacking**

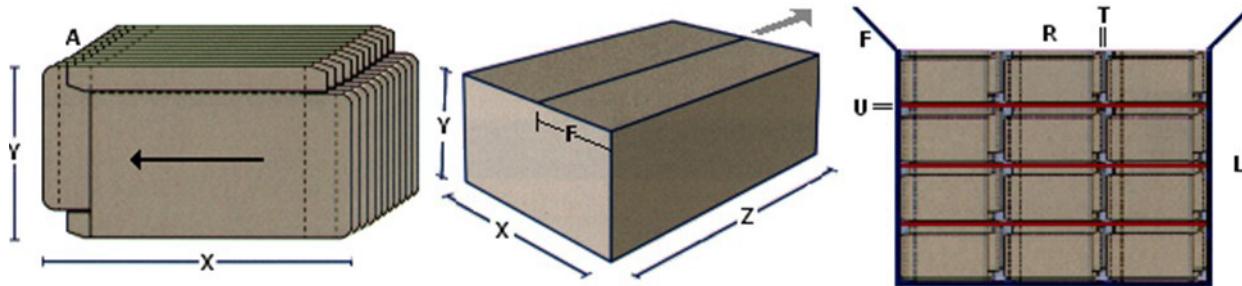
Output of Processes: —

Table 8-16: BoxPackingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>BoxType</i> ?	enumeration	<p>@<i>BoxType</i> specifies the general category of the package to be packed.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Box</i> – Boxes are convenience packaging and are not envisioned to be protection for shipping. <i>Carton</i> – Cartons envisioned to be protection for shipping. <i>Envelope</i> – Envelopes are packages that are envisioned for shipping. <i>Roll</i> – Rolls are cylinder shaped cartons that are envisioned for shipping.
<i>BoxTypeDetails</i> ?	string	<p>Additional details of @<i>BoxType</i>. @<i>BoxType</i> MAY be a site specific identifier.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>NeutralCarton</i> <i>BrandedCarton</i> <i>EasterBunnyBox</i>
<i>ComponentsPerRow</i> ?	integer	Components per row in the shipping box, as illustrated by A in Figure 8-13. If the Components represent Bundles , the number of Bundles is specified.
<i>Columns</i> ?	integer	Columns per shipping box. Columns are in the 3rd Dimension in Figure 8-13, and are thus not illustrated.
<i>ComponentOrientation</i> ?	enumeration	<p>Defines the coordinate pair that is facing the bottom of the box, defining the horizontal plane.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>XY</i> – Axis X and Y <i>XZ</i> – Axis X and Z <i>YZ</i> – Axis Y and Z
<i>Copies</i> ?	integer	Number of copies in the box. @ <i>Copies</i> SHALL NOT be specified if @ <i>MaxWeight</i> is present.

Table 8-16: BoxPackingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>FillMaterial</i> ?	NMTOKEN	Material to fill boxes that are not completely filled, as illustrated by F in Figure 8-13. Values include: <i>Any</i> – Explicit request for system specified filling. <i>BlisterPack</i> <i>None</i> – Explicit request for no filling. <i>Paper</i> <i>Styrofoam</i>
<i>Layers</i> ?	integer	Layers per shipping box, as illustrated by L in Figure 8-13.
<i>MaxWeight</i> ?	double	Maximum weight of a packed box in grams. @ <i>MaxWeight</i> SHALL NOT be specified if @ <i>Copies</i> is present.
<i>Pattern</i> ?	NMTOKEN	Name of the box packing pattern. Used to store a predefined pattern that defines the layers and positioning of individual component in the box or carton.
<i>Rows</i> ?	integer	Rows per shipping box, as illustrated by R in Figure 8-13.
<i>Ties</i> ?	IntegerList	Number of tie Sheets at each row. The first value is outside the first row, the next value between the first and second row and so forth. If more rows than values are specified, counting restarts at the 0 position. If fewer layers than values are specified, all tie Sheets that are not adjacent to a row are ignored.
<i>UnderLays</i> ?	IntegerList	Number of underlay Sheets at each layer, as illustrated by U in Figure 8-13. The first value is underneath the bottom layer, the next value above the first layer and so forth. If more layers than values are specified, counting restarts at the 0 position. If less layers than values are specified, all underlay Sheets that are not adjacent to a layer are ignored.

Figure 8-10: Box packing

8.12 BufferParams

This Resource provides controls for **Buffer** Process.

Resource Properties

Resource referenced by: —

Input of Processes: **Buffer**

Output of Processes: —**Table 8-17: BufferParams Resource**

Name	Data Type	Description
<i>MinimumWait</i> ?	duration	Minimum amount of time that an individual Resource SHALL be buffered.

8.13 Bundle

Bundles are used to describe various kinds of sets of sheets. Note that **Bundle** Resources MAY be created by many press or postpress Processes and not only **Bundling**.

Resource Properties

Resource referenced by:

Input of Processes: —

Output of Processes: —

Table 8-18: Bundle Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>BundleType</i> ?	enumeration	Allowed values are: <i>BoundSet</i> – Stack of components that are bound together. <i>Box</i> <i>Carton</i> <i>CollectedStack</i> – Components collected on a saddle, result of Collecting Process <i>CompensatedStack</i> – Loose stack of compensated components <i>Pallet</i> <i>Roll</i> – Rolled components on a print Roll. <i>Sheet</i> – Multiple individual items printed onto one Sheet. <i>SheetStream</i> – Stream of individual sheets that are continuously moved from one device to another (e.g., in an inline digital finishing device). <i>Stack</i> – Loose stack of equally stacked components. <i>StrappedStack</i> – Strapped stack of equally stacked components. <i>StrappedCompensatedStack</i> – Strapped stack of compensated components. <i>WrappedBundle</i>
<i>SheetCount</i> ?	integer	Total number of physical sheets that this Bundle contains.
<i>TotalAmount</i> ?	integer	Total amount of individual products that this bundle contains. If the bundle contains one or more [contains (@ <i>ComponentType</i> , "Final Product")], @ <i>TotalAmount</i> refers to the number of final products. Note that this is neither always the next level of <i>BundleItem</i> nor the lowest level of <i>BundleItem</i> . For instance, the next level MAY be the boxes in a carton, whereas the lowest level MAY be the Sheets comprising the brochure. The correct number in this example would be the number of Brochures. If not specified, it SHALL be calculated from the individual <i>BundleItem</i> Elements.

Table 8-18: Bundle Resource (Sheet 2 of 2)

Name	Data Type	Description
BundleItem *	element	References to the individual items that form this Bundle .

8.13.1 Element: BundleItem

A **Bundle** is described as a set of BundleItem Elements. Since BundleItem Elements reference Resources which themselves can reference further **Bundle** Resources, the structure is recursive.

Table 8-19: BundleItem Element

Name	Data Type	Description
Amount	integer	Number of this type of items.
Ref	IDREF	Reference to a that is part of this Bundle .
Orientation ?	Orientation	Named Orientation of the respective to the Bundle coordinate system. For details, see Table 2-4, “Matrices and Orientation values for describing the orientation of a Component” on page 39. At most one of @Orientation or @Transformation SHALL be specified.
Transformation ?	matrix	Orientation of the respective to the Bundle coordinate system. At most one of @Orientation or @Transformation SHALL be specified.

Example 8-4: Bundle: Boxing and Palletizing

The following example code shows an XJDF that describes boxing and palletizing for 4200 books. The appropriate **Bundle** Elements have orange tags and magenta Attributes. The Resources have not yet been completely filled in.

TBD 2.x Example.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="Bundle" Status="Waiting"
      Type="ProcessGroup" JobPartID="ID20" Version="1.4">
  <!-- The BoxPacking Process consumes the thing to pack and the boxes-->
  <!-- The BoxPacking Process creates packed boxes -->
  <JDF ID="n0235" Status="Waiting" Type="BoxPacking" JobPartID="ID21" >
    <ResourceLinkPool>
      <ComponentLink ProcessUsage="Box" Usage="Input" rRef="BoxID"/>
      <BoxPackingParamsLink Usage="Input" rRef="BoxParamsID"/>
      <ComponentLink Usage="Input" rRef="ComponentID"/>
      <ComponentLink Usage="Output" rRef="PackedBoxID"/>
    </ResourceLinkPool>
    <!-- The BoxPacking Process has the following local resources -->
    <ResourcePool>
      <BoxPackingParams Class="Parameter" ID="BoxParamsID"
                        Status="Available"/>
      <Component Amount="100" Class="Quantity" ID="BoxID"
                 Status="Available" ComponentType="Sheet"/>
    </ResourcePool>
  </JDF>
<ResourcePool>
  <!-- This Component describes a Box with 42 Books -->
  <Component Amount="100" Class="Quantity" ID="PackedBoxID"
             Status="Unavailable" ComponentType="Sheet" >
    <Bundle BundleType="Box" TotalAmount="42">
      <BundleItem Amount="42">
        <ComponentRef rRef="ComponentID"/>
      </BundleItem>
    </Bundle>
  </Component>
```

```

<Component Amount="4200" Class="Quantity" ID="ComponentID"
    Status="Available" ComponentType="Sheet" />
    <!-- This Component describes the contents of the pallet: 100
        Boxes w. 42 Books -->
<Component Amount="10" Class="Quantity" ID="palletContentsID"
    Status="Unavailable" ComponentType="Sheet" >
    <Bundle BundleType="Pallet" TotalAmount="420">
        <BundleItem Amount="10">
            <ComponentRef rRef="PackedBoxID"/>
        </BundleItem>
    </Bundle>
</Component>
</ResourcePool>
<JDF ID="n0239" Status="Waiting" Type="Palletizing" JobPartID="ID22">
    <ResourceLinkPool>
        <ComponentLink Usage="Input" rRef="PackedBoxID"/>
        <PalletLink Usage="Input" rRef="palletID"/>
        <PalletizingParamsLink Usage="Input" rRef="palletParamsID"/>
        <ComponentLink Usage="Output" rRef="palletContentsID"/>
    </ResourceLinkPool>
    <ResourcePool>
        <Pallet Amount="10" Class="Consumable" ID="palletID"
            Status="Available" PalletType="Euro800x600"/>
        <PalletizingParams Class="Parameter" ID="palletParamsID"
            Status="Available" />
    </ResourcePool>
</JDF>
</JDF>

```

8.14 BundlingParams

BundlingParams describes the details of a **Bundling** Process.

Resource Properties

Resource references: —

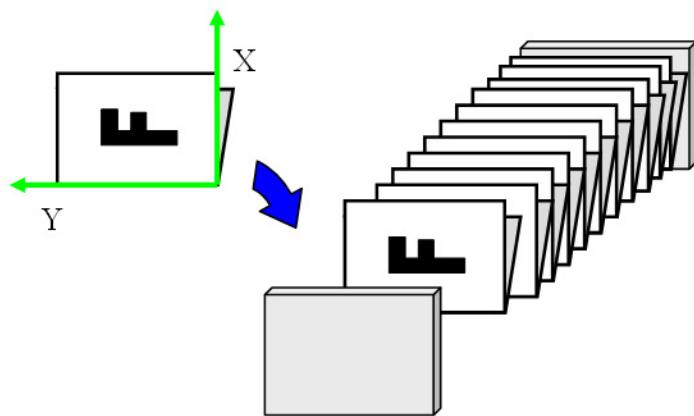
Input of Processes: **Bundling**

Output of Processes: —

Table 8-20: BundlingParams Resource

Name	Data Type	Description
<i>Copies</i> ?	integer	Number of products within a bundle. @ <i>Copies</i> SHALL NOT be specified if @ <i>Length</i> is present.
<i>Length</i> ?	double	Length of a bundle. @ <i>Length</i> SHALL NOT be specified if @ <i>Copies</i> is present.

Figure 8-11: BundlingParams Coordinate System



8.15 CaseMakingParams

This Resource describes the settings of a *CaseMaking* Process.

Resource Properties

Resource referenced by:

—

Intent Pairing:

BindingIntent

Input of Processes:

CaseMaking

Output of Processes:

—

Figure 8-12: CaseMakingParams

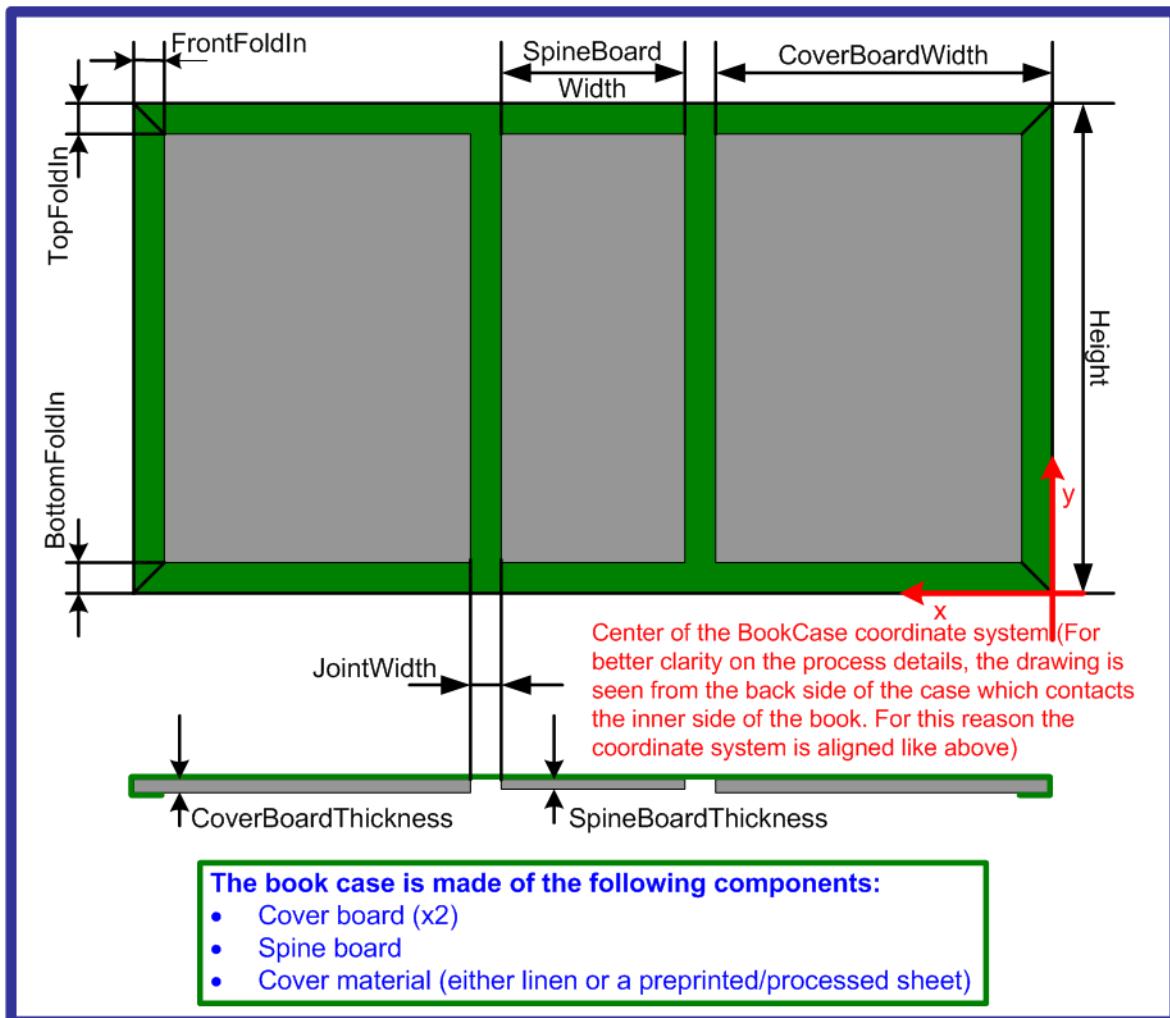


Table 8-21: CaseMakingParams Resource (Sheet 1 of 2)

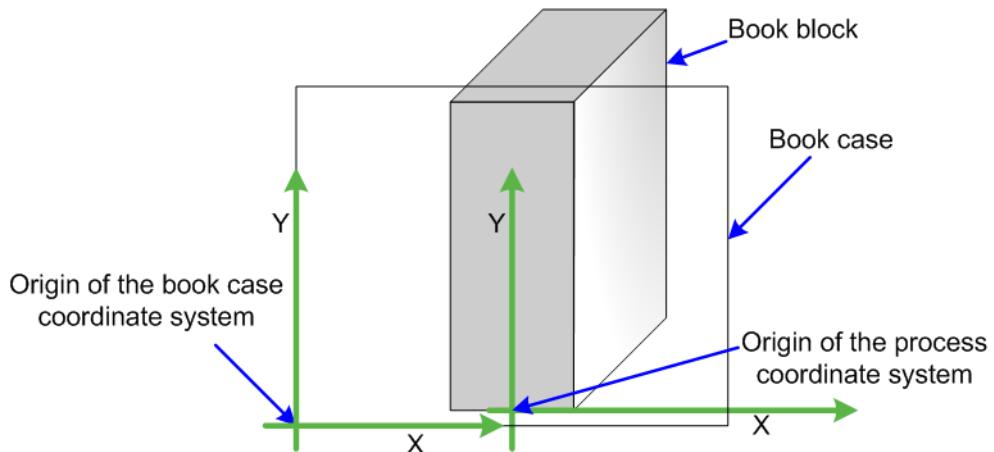
Name	Data Type	Description
<i>BottomFoldIn</i> ?	double	Defines the width of the part of the CoverMaterial on the lower edge inside of the case. If not specified, defaults to <i>@TopFoldIn</i> .
<i>CornerType</i> ?	NMTOKEN	Method of wrapping the corners of the cover material around the corners of the board. Values include: <i>LibraryCorner</i> – The American Library Corner style.
<i>CoverWidth</i> ?	double	Width of the cover cardboard in points.
<i>FrontFoldIn</i> ?	double	Defines the width of the part of the cover material on the front edges inside of the case.
<i>Height</i> ?	double	Height of the book case, in points.
<i>JointWidth</i> ?	double	Width of the joint as seen when laying the cardboard on the cover material, in points.

Table 8-21: CaseMakingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>SpineWidth</i> ?	double	Width of the spine cardboard, in points.
<i>TopFoldIn</i> ?	double	Defines the width of the cover material on the top edge inside of the case.
<i>GlueLine</i> ?	element	Details of the glue. Because the glue is applied to the whole back side of the cover material, <i>GlueLine/@AreaGlue</i> SHALL be set to "true".

8.16 CasingInParams

This Resource describes the settings of a **CasingIn** Process. The geometry is always centered See Figure 8-16.

Figure 8-13: Parameters and coordinate system for CasingIn

Resource Properties

Resource referenced by: —

Intent Pairing: **BindingIntent**

Input of Processes: **CasingIn**

Output of Processes: —

Table 8-22: CasingInParams Resource

Name	Data Type	Description
<i>CaseRadius</i> ?	double	Inner radius of the case spine rounding. If not specified, no rounding of the case spine is performed.
<i>CoverBoardWidth</i> ?	double	Width of the CoverBoard. Note that Height and total Case Dimensions are specified in the Component (Case) of the CasingIn process. For details of <i>@CoverBoardWidth</i> , see also Figure 8-15, "CaseMakingParams," on page 558.
<i>SpineBoardWidth</i> ?	double	Width of the SpineBoard. Note that Height and total Case Dimensions are specified in the Component (Case) of the CasingIn process. For details of <i>@SpineBoardWidth</i> , see also Figure 8-15, "CaseMakingParams," on page 558.
<i>GlueLine</i> *	element	Properties of the glue to attach the case.

8.17 CollectingParams

The **Collecting** Process needs no special Attributes. However, this Resource is provided as a container for extensions of the **Collecting** Process.

Resource Properties

Resource referenced by: —

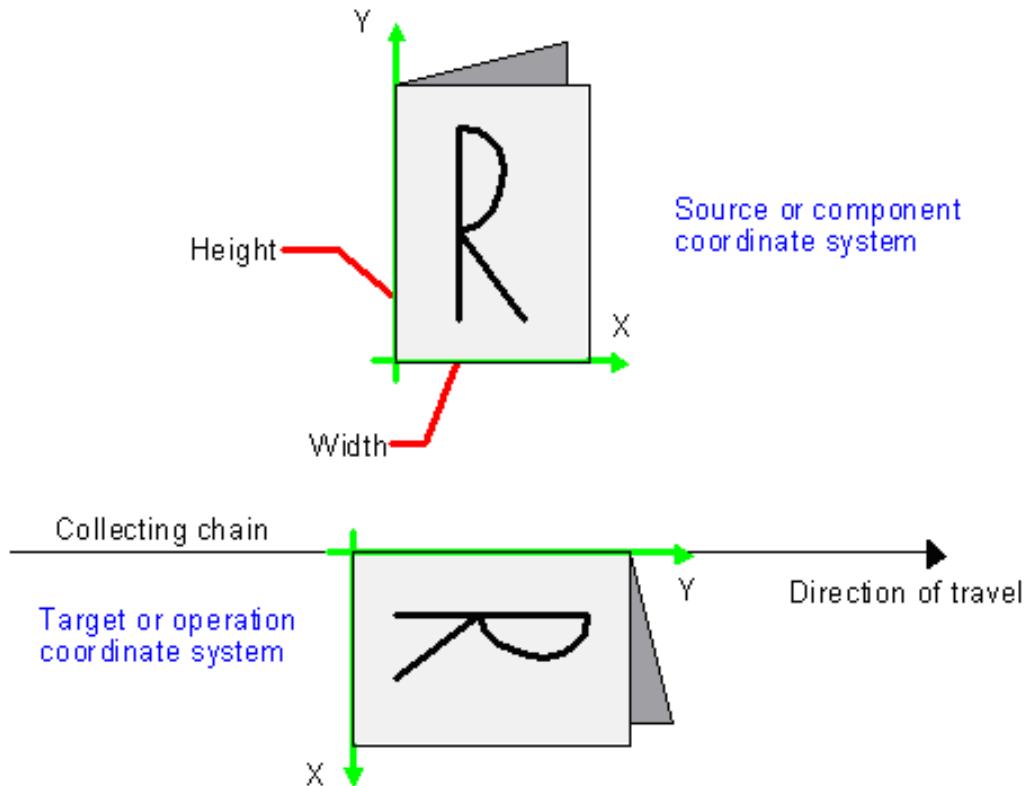
Input of Processes: **Collecting**

Output of Processes: —

Table 8-23: CollectingParams Resource

Name	Data Type	Description

Figure 8-14: Coordinate systems used for collecting



8.18 Color

Color describes the details of spot color inks, process color inks and any other coating, for instance varnish or gloss coating. Spot colors are named colors that can either be separated or converted to process colors. It is important to know the neutral density of the colorant for trapping and, in many cases, the *@Lab* values for representing them on screen. If you know the *@Lab* value, you can calculate the neutral density. At most one *ResourceSet[@Name="Color"]* SHALL be specified in one XJDF instance. This Color *ResourceSet* summarizes the properties of all colorants that are used in the XJDF.

A color is represented by a **Color** Element. When **ColorantAlias** has been used in **ColorantControl** to clean up string names of spot colors, the resolved, not the uncorrected duplicate, **ColorantAlias/@ReplacementColorantName** spot color name SHALL match **Color/../Resource/Part[@Separation]**. The four names that are reserved for representing process CMYK color names are "*Cyan*", "*Magenta*", "*Yellow*" and "*Black*". Every colorant MAY have a **@Lab** and/or **@CMYK** color value. If both are specified and a system is capable of interpreting both values, the **@Lab** value overrides the **@CMYK** definition, unless the target Device is compatible with CMYK (i.e., **ColorantControl/@ProcessColorModel = "DeviceCMYK"**). In this case the CMYK value has precedence.

The **@Lab** value represents the *L*, *a*, *b* readings of the ink on certain media. This means that spot inks printed on three different kinds of stocks have different **@Lab** values. Pantone books, for example, provide **@Lab** values for three kinds of paper: "*Coated*" (not necessarily glossy), "*Matte*" and "*Uncoated*". Thus a color of ink SHOULD identify the media for which the Color is specified. CMYK colors are used to approximate spot colors when they are not separated. This conversion can be done by a color management system, or there can be fixed CMYK representation defined by color books such as Pantone.

Resource Properties

Resource referenced by: —

Intent Pairing: **ColorIntent**

Input of Processes: —

Output of Processes: —

Table 8-24: Color Resource (Sheet 1 of 4)

Name	Data Type	Description
<i>ActualColorName</i> ?	string	Actual name of the color in the PDL. @ActualColorName SHOULD be used to identify the color. If not specified, defaults to the value of @Name .
<i>CMYK</i> ?	CMYKColor	CMYK value of the 100% tint value of the colorant. Although OPTIONAL, it is highly RECOMMENDED that this value be filled when the colorant is a spot colorant (i.e., not part of the @ProcessColorModel). This preferred CMYK MAY be associated with an ICC source profile defined in the FileSpec Resource with a @ResourceUsage = "ColorProfile" when the target CMYK is different from the PDL CMYK.
<i>ColorBook</i> ?	string	Definition of the color identification book name that is used to represent this color. The color book name SHALL match the name defined by the color book vendor Values include: <i>CIP4 ColorBook Uncoated Grade 5</i> <i>PANTONE C – an example</i> <i>PANTONE C – an example</i>

Table 8-24: Color Resource (Sheet 2 of 4)

Name	Data Type	Description
<i>ColorBookEntry</i> ?	string	Definition of the Color within the standard specified by <i>@ColorBook</i> . This entry SHALL exactly match the color book entry as defined by the <i>@ColorBook</i> specified vendor, including capitalization and media type extension. When using ICC Profiles, this maps to the NCL2 value of a namedColorType tag of an ICC color profile. This entry is used to map from the XJDF Color to an ICC namedColorType tag.
<i>ColorBookPrefix</i> ?	string	Definition of the name prefix of the color book entry within a named ICC profile. This entry is used to map from the XJDF Color to an ICC namedColorType tag.
<i>ColorBookSuffix</i> ?	string	Definition of the name suffix of the color book entry within a named ICC profile. This entry is used to map from the XJDF Color to an ICC namedColorType tag.
<i>ColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If <i>@ColorDetails</i> is supplied, <i>@ColorName</i> SHOULD also be supplied.
<i>ColorName</i> ?	NamedColor	Mapping to a color name.
<i>ColorType</i> ?	enumeration	<p>A name that characterizes the colorant.</p> <p>Allowed values are:</p> <p><i>DieLine</i> – Marks made with colorants of this type are ignored for trapping. Trapping Processes need not generate a color plane for this colorant. "<i>DieLine</i>" can be used for auxiliary process separations. "<i>DieLine</i>" marks will generally appear on proof output but will not be marked on final output (e.g., plates). Note that the ColorantControl Resource SHALL be correctly set up for the RIP and that <i>@ColorType = "DieLine"</i> does not implicitly remove the "<i>DieLine</i>" separation from final output.</p> <p><i>Normal</i> – Marks made with colorants of this type, marks covered by colorants of this type, and marks on top of colorants of this type are trapped.</p> <p><i>Transparent</i> – Marks made with colorants of this type are to be ignored for trapping. Trapping Processes are not to generate a color plane for this colorant. This value SHOULD be used for varnish.</p> <p><i>Opaque</i> – Marks covered by colorants of this type are ignored for trapping. "<i>Opaque</i>" can be used for metallic inks.</p> <p><i>OpaqueIgnore</i> – Marks made with colorants of this type and marks covered by colorants of this type are ignored for trapping. "<i>OpaqueIgnore</i>" can be used for metallic inks.</p>

Table 8-24: Color Resource (Sheet 3 of 4)

Name	Data Type	Description
<i>ColorTypeDetails</i> ?	string	Additional information about the color type. If <i>@ColorType</i> = "DieLine", this attribute SHOULD specify the type of die line (e.g., DDES-numbers, For details, see Table 8-33, "Diecutting Data (DDES3)" on page 569 for a list of DDES3 die line types.
<i>Density</i> ?	double	Density value of colorant (100% tint). Whereas <i>@NeutralDensity</i> describes measurements of inks on substrate with wide-band filter functions, <i>@Density</i> is derived from measurements of inks on substrate with special small-band filter functions according to ANSI and DIN.
<i>Gray</i> ?	double	Gray value of the 100% tint value of the colorant. Although OPTIONAL, it is highly RECOMMENDED that this value be filled when the colorant is a spot colorant, <i>@MappingSelection</i> = "UseProcessColorValues" and <i>ColorantControl/@ProcessColorModel</i> = "DeviceGray". Uses a subtractive color model: 0.0 means 100% coverage with colorant, while 1.0 means no coverage.
<i>Lab</i> ?	LabColor	L, a, b value of the 100% tint value of the colorant.
<i>MappingSelection</i> ?	enumeration	<p>This value specifies the mapping method to be used for this Color.</p> <p><i>@MappingSelection</i> can be specifically used to indicate how a combination of process colorant values will be obtained for any spot color when the separation spot colorant itself is not to be used.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>UsePDLValues</i> – Use color values specified in the PDL for this color. See [ColorPS]. <i>UseLocalPrinterValues</i> – Use the Printer's best local mapping for this Color. <i>UseProcessColorValues</i> – Use the values defined in this Color.
<i>MediaType</i> ?	NMTOKEN	<p>Specifies the media type.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Coated</i> – Pertains to gloss coated. <i>Matte</i> – Pertains to matte or dull coated. <i>Uncoated</i>
<i>NeutralDensity</i> ?	double	A number in the range of 0.001 to 10 that represents the neutral density of the colorant, defined as $10 * \log(1/Y)$. Y is the tristimulus value in CIEXYZ coordinates, normalized to 1.0.

Table 8-24: Color Resource (Sheet 4 of 4)

Name	Data Type	Description
<i>PrintingTechnology</i> ?	NMTOKEN	<p>Printing technology of the press, press module or printer. For digital printing, describes the printing technology that the media or coatings on the media are intended for or optimized for.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>DyeSublimation</i> – for digital printing. <i>Electrostatic</i> – for digital printing. <i>Flexo</i> – for conventional printing. <i>Gravure</i> – for conventional printing. <i>InkJet</i> – for digital printing. <i>Laser</i> – for digital printing. <i>Offset</i> – for digital and conventional printing. <i>Screen</i> – for conventional printing. <i>Thermal</i> – for digital printing.
<i>RawName</i> ?	hexBinary	Representation of the original 8-bit byte stream of the Color @Name . Used to transport the original byte representation of a Color @Name when moving XJDF tickets between computers with different locales. Only one Color with any given @RawName SHALL be specified in a ResourceSet[@Name="Color"] .
<i>sRGB</i> ?	sRGBColor	sRGB value of the 100% tint value of the colorant.
<i>DeviceNColor</i> *	element	Each DeviceNColor element defines the colorant in the DeviceN color space that is selected by DeviceNColor/@Name .
FileSpec (ColorProfile) ?	element	A FileSpec Resource pointing to an ICC named color profile that describes further details of the color. This ICC profile is intended as a source profile for the named color whose equivalent CMYK value is given in the @CMYK Attribute.
FileSpec (TargetProfile) ?	element	A FileSpec Resource pointing to an ICC profile that defines the target output Device in case the object that uses the Color has been color space converted to a Device color space. FileSpec (TargetProfile) applies to the alternate color defined by the value of @MappingSelection .

8.18.1 Element: **DeviceNColor**

Table 8-25: DeviceNColor Element (Sheet 1 of 2)

Name	Data Type	Description
<i>ColorList</i>	DoubleList	Value of the 100% tint value of the colorant in the DeviceNSpace that is selected by @Name . Each index SHALL refer to the separation defined at the same position in DeviceNSpace/@Separations . The number of entries SHALL be DeviceNSpace/@N . A value of 0 specifies no ink and a value of 1 specifies full ink.

Table 8-25: DeviceNColor Element (Sheet 2 of 2)

Name	Data Type	Description
Name	NMTOKEN	@Name of the matching DeviceNSpace. Exactly one ColorantControl/DeviceNSpace/@Name SHALL match @Name.

8.18.2 Diecutting Data (DDES3)

The following list of line types is taken from Annex A of ANSI® IT8.6-2002 Graphic technology — Prepress digital data exchange — Diecutting data (DDES3). The list is included in the XJDF specification with permission of IT8.6. A full copy of the DDES3 standard can be obtained from [http://webstore.ansi.org/RecordDetail.aspx?sku=ANSI+IT8.6-2002+\(R2013\)](http://webstore.ansi.org/RecordDetail.aspx?sku=ANSI+IT8.6-2002+(R2013)).

Table 8-26: Diecutting Data (DDES3)

DDES 3 Line type number	DDES3 Line type	Description
12	Non-varnish / UV area	Contour indicating a varnish free area
15	Printing / UV Blanket Edge	Contour enclosing a spot varnish area. Spot varnish will be applied with a varnish blanket.
16	Zipper / Tear Strip / Tear Edge - reference lines for cutting edge	Cutting contours indicating a tear strip.
17	Wave / Scallop - reference lines for cutting edge	Cutting contours indicating a wave /scallop (note: I have no clue what this is...)
18	Punches - reference lines for center / cutting edge	Contours indicating the shape and center of a punch
100	Miscellaneous ruled lines for dies	
101	Knife / Cutting rule	Contour indicating how the printed artwork will be cut from the printed sheet e.g. with a guillotine cutter or die cutting device.
102	Crease / Scoring rule	Contour indicating where the substrate will be creased to guide subsequent folding.
103	Perforation (Alternating cutting and spaces)	Contour indicating where the substrate will be perforated.
104	Cutscore / Halfcut (Partial depth cutting rule)	Contour indicating where the substrate will be cut partially i.e. not entirely through the material. Cutting is done from the front side.
105	Cut-Crease rule (Alternating cutting and creasing rule)	Contour indicating alternating cutting and creasing
106	Cutscore-Crease (Alternating partial depth cutting and creasing rule)	Contour indicating alternating half-cutting and creasing
107	Reverse cutscore / halfcut (for anvil in die)	Contour indicating where the substrate will be cut partially i.e. not entirely through the material. Cutting is done from the back side.
108	Emboss / Deboss crease profile	Contour enclosing an area where embossing will be applied.

Example 8-5: Color

This is an example of the structure for **Color**. The transfer curves in this example are defined for process CMYK and sRGB, independently.

TBD 2.x Example.

```
<Color Class="Parameter" ID="C000" Status="Available" CMYK="0.2 0.3 0.4 0.5"
      Density="3.14" Lab="20. 30. 40." MediaType="Coated"
      Name="PANTONE Deep Blue" sRGB="0.6 0.7 0.9">
  <TransferCurve Curve="0 0 .5 .4 1 1" Separation="Cyan"/>
  <TransferCurve Curve="0 0 .5 .6 1 1" Separation="Magenta"/>
  <TransferCurve Curve="0 0 1 1" Separation="Yellow"/>
  <TransferCurve Curve="0 0 1 1" Separation="Black"/>
  <TransferCurve Curve="0 0 1 1" Separation="sRed"/>
  <TransferCurve Curve="0 0 1 1" Separation="sGreen"/>
  <TransferCurve Curve="0 0 1 1" Separation="sBlue"/>
</Color>
```

Example 8-6: ColorantControl: Content-Ignorant MIS

TBD 2.x Example.

```
<ColorantControl Class="Parameter" ID="r000004" ProcessColorModel="DeviceCMYK"
                  Status="Available">
  <!--Note that all Strings in ColorantParams etc. use Color/@Name,
      NOT Color/@ActualColorName-->
  <ColorantParams>
    <SeparationSpec Name="Spot1"/>
    <SeparationSpec Name="BlackText"/>
  </ColorantParams>
</ColorantControl>
```

Example 8-7: ColorantControl: Synchronized with Input

Example of initial (previous) **ColorantControl** after synchronizing with input. This example specifies the replacement color name with a new *@ActualColorName* Attribute in the **Color** Element.

TBD 2.x Example.

```
<!--ColorantControl after prepress has correctly set ActualColorName based
    on pdl content-->
<ColorantControl Class="Parameter" ID="r000004" ProcessColorModel="DeviceCMYK"
                  Status="Available">
  <!--Note that all Strings in ColorantParams etc. use Color/@Name,
      NOT Color/@ActualColorName-->
  <ColorantParams>
    <SeparationSpec Name="Spot1"/>
    <SeparationSpec Name="BlackText"/>
  </ColorantParams>
  <ColorPoolRef rRef="r000005"/>
</ColorantControl>
<ColorPool Class="Parameter" ID="r000005" Status="Available">
  <!--Color that maps the predefined separation Black
      ActualColorName is the new attribute that replaces
      ExposedMedia/@DescriptiveName as the "Main" PDL color
      -->
  <Color ActualColorName="Schwarz" CMYK="0 0 0 1" Class="Parameter"
        Name="Black"/>
  <Color ActualColorName="Gelb" CMYK="0 0 1 0" Class="Parameter"
        Name="Yellow"/>
```

```

<!--ActualColorName defaults to Name if not specified-->
<Color CMYK="1 0 0 0" Class="Parameter" Name="Cyan"/>
<Color Class="Parameter" Name="Magenta"/>
<Color ActualColorName="Acme Aqua" CMYK="0.7 0.2 0.03 0.1"
      Class="Parameter" Name="Spot1"/>
<Color ActualColorName="VersionsText" CMYK="0 0 0 1" Class="Parameter"
      Name="BlackText"/>
</ColorPool>

```

Example 8-8: ColorantControl: Synchronized with Input with Alias

Example of initial **ColorantControl** after synchronizing with input that contains an alias

TBD 2.x Example.

```

<ColorantControl Class="Parameter" ID="r000004" ProcessColorModel="DeviceCMYK"
    Status="Available">
    <!--ColorantControl after prepress has correctly set ActualColorName based
        on pdl content-->
    <!--Note that all Strings in ColorantParams etc. use Color/@Name,
        NOT Color/@ActualColorName-->
    <ColorantParams>
        <SeparationSpec Name="Spot1"/>
        <SeparationSpec Name="BlackText"/>
    </ColorantParams>
    <ColorPoolRef rRef="r000005"/>
    <!--ColorantAlias that maps the additional representations
        (noir, schwarz) to the predefined separation Black-->
    <ColorantAlias Class="Parameter" RawNames="6E6F6972 73636877E4727A"
        ReplacementColorantName="Black">
        <SeparationSpec Name="noir"/>
        <SeparationSpec Name="schwarz"/>
    </ColorantAlias>
</ColorantControl>
<ColorPool Class="Parameter" ID="r000005" Status="Available">
    <!-- ColorPool is same as previous example -->
</ColorPool>

```

Example 8-9: ColorantControl: with ColorantAlias/ReplacementColorantName

Example of many-to one substitution with **ColorantAlias/@ReplacementColorantName**

TBD 2.x Example.

```

<ColorantControl Class="Parameter" ID="r000004" ProcessColorModel="DeviceCMYK"
    Status="Available">
    <!--ColorantAlias that maps the predefined separation Black-->
    <ColorantAlias ReplacementColorantName="Black">
        <SeparationSpec Name="Schwarz"/>
        <SeparationSpec Name="schwarz"/>
    </ColorantAlias>
</ColorantControl>

<ColorantControl Class="Parameter" ID="r000004" ProcessColorModel="DeviceCMYK"
    Status="Available">
    <!--ColorantAlias that maps the predefined separation Black-->
    <ColorantAlias ReplacementColorantName="Black">
        <SeparationSpec Name="Schwarz"/>
    </ColorantAlias>
    <ColorantAlias ReplacementColorantName="Black">
        <SeparationSpec Name="schwarz"/>
    </ColorantAlias>

```

```
</ColorantAlias>
</ColorantControl>
```

8.19 ColorantControl

ColorantControl defines how color separations of PDL or raster data SHALL be output on a target device. Color separations can be either output as real colorants, mapped to process colorants or ignored.

Colorants are referenced in **ColorantControl** by matching values of **Part/@Separation**. Additional details about individual colorants can be found in **ResourceSet[@Name="Color"]**.

Resource Properties

Resource referenced by:

—
ColorIntent

Intent Pairing:

ColorCorrection, ColorSpaceConversion, ConventionalPrinting, DigitalPrinting, ImageSetting, Interpreting, PreviewGeneration, Separation, Stripping, Trapping

Input of Processes:

ColorSpaceConversion

Output of Processes:

Table 8-27: ColorantControl Resource (Sheet 1 of 3)

Name	Data Type	Description
<i>ColorantConvertProcess</i> ?	NMTOKENS	List of colors that SHALL be converted to process colors. Defaults to all colors that are not listed in @ColorantParams . An Application MAY issue a warning for all PDL Colors that are not in (@ColorantParams + @ColorantConvertProcess) .
<i>ColorantOrder</i> ?	NMTOKENS	The ordering of named colorants to be processed, for example in the RIP. All of the colorants named SHALL occur in the @ColorantParams list. If present, then only the colorants specified by @ColorantOrder SHALL be output. Colorants listed in the @ColorantParams , but not listed in @ColorantOrder , SHALL NOT be output. They SHALL still be processed for side effects in the colorants that are listed such as knockouts or trapping. If not present, then all colorants specified in @ColorantParams are output.
<i>ColorantParams</i> ?	NMTOKENS	A set of named colorants. This list defines all the colorants that are expected to be available on the Device where the Process will be executed. Named colors found in the PDL that are not listed in @ColorantParams will be implemented through their @ProcessColorModel equivalents. (See ColorSpaceConversion Process.) The colorants implied by the value of @ProcessColorModel SHALL be specified in this list. The spot colors defined in ColorIntent/@ColorsUsed will in general be mapped to @ColorantParams for each spot color to be used as part of any Product Intent to Process conversion.

Table 8-27: ColorantControl Resource (Sheet 2 of 3)

Name	Data Type	Description
<i>ForceSeparations</i> ?	boolean	If "true", forces all colorants to be output as individual separations, regardless of any values defined in ColorantControl (i.e., all separations in a document are assumed to be valid and are output individually). A value of "false" specifies to respect the parameters specified in ColorantControl and elsewhere in the XJDF .
<i>InternalColorModel</i> ?	enumeration	<p>Internal color model that SHALL be used by a device that supports enhanced color models.</p> <p>Allowed values are:</p> <p><i>Basic</i> – Use the basic color model selected by this ColorantControl.</p> <p><i>Enhanced</i> – Use the enhanced color model that is implied by this ColorantControl (e.g., Use "<i>LightCyan</i>", "<i>LightMagenta</i>" in addition to CMYK).</p> <p><i>Explicit</i> – Use the elements of the enhanced color model that are explicitly listed in <i>@ColorantOrder</i>.</p>
<i>MappingSelection</i> ?	enumeration	<p>This value specifies the default mapping method to be used for all separations. Note that <i>@MappingSelection</i> MAY be overridden by Color/<i>@MappingSelection</i>.</p> <p><i>@MappingSelection</i> can be specifically used to indicate how a combination of process colorant values will be obtained for any spot color when the separation spot colorant itself is not to be used.</p> <p>Allowed values are:</p> <p><i>UsePDLValues</i> – Use color values specified in the PDL for a color. See [ColorPS].</p> <p><i>UseLocalPrinterValues</i> – Use the Printer's best local mapping for a Color.</p> <p><i>UseProcessColorValues</i> – Use the values defined in the respective Color.</p>
<i>ProcessColorModel</i> ?	enumeration	<p>Specifies the model to be used for rendering the colorants defined in color spaces into process colorants.</p> <p>Allowed values are:</p> <p><i>DeviceCMY</i></p> <p><i>DeviceCMYK</i></p> <p><i>DeviceGray</i></p> <p><i>DeviceN</i> – The specific DeviceN color space to operate on is defined in the DeviceNSpace Resource. If this value is specified then DeviceNSpace SHALL also be present.</p> <p><i>DeviceRGB</i></p> <p><i>None</i> – No Colorants other than those specified in <i>@ColorantParams</i> SHALL be output.</p> <p>Note: The separations implied by <i>@ProcessColorModel</i> SHALL be explicitly specified in <i>@ColorantConvertProcess</i>.</p>

Table 8-27: ColorantControl Resource (Sheet 3 of 3)

Name	Data Type	Description
ColorantAlias *	element	Identify one or more named colorants that are to be replaced with a specified named colorant.
DeviceNSpace ?	element	DeviceNSpace defines the colorants that make up a DeviceN color space. The DeviceNSpace Element SHALL be present if the <i>@ProcessColorModel</i> value is "DeviceN".

The following table describes which separations are output for various values of *@ProcessColorModel*, *@ColorantOrder*, **ColorantControl**, *@ColorantParams* and *@DeviceColorantOrder*. Note that all separations that are neither specified in *@ColorantParams* nor implied by *@ProcessColorModel* are mapped to the colors implied by *@ProcessColorModel* prior to any color selection defined by *@ColorantOrder*.

Table 8-28: Sample output for different values of ProcessColorModel, ColorantParams, ColorantOrder, ColorantControlLink and DeviceColorantOrder Elements.

ProcessColorModel	ColorantParams	ColorantOrder	Colorants not shown in the output	Separations that are output and ordered for press using DeviceColorantOrder
DeviceCMYK	Not Present	Cyan Magenta	Yellow Black	Cyan Magenta (If <i>@DeviceColorantOrder</i> is not present then lay down order will be Cyan first, Magenta last.)
DeviceCMYK	Spot1 Spot2	Cyan Magenta Yellow Black Spot2	Spot1	Cyan Magenta Yellow Black Spot2
DeviceGray	Spot1 Spot2	Black Spot2	Spot1	Black Spot2
DeviceN (with example N = 2 colorants as identified in DeviceNSpace)	Spot1 Spot2	Spot2 DeviceN 1 DeviceN 2	Spot1	DeviceN 1 DeviceN 2 Spot2 The reordering is accomplished using <i>@DeviceColorantOrder</i>

8.19.1 Element: DeviceNSpace

DeviceNSpace lists the process color separations that define a non-standard process color space. Additional details of the DeviceNSpace SHOULD be provided as references to ICC profiles, e.g. in ColorSpaceConversionParams/FileSpec(*FinalTargetDevice*).

Table 8-29: DeviceNSpace Element

Name	Data Type	Description
<i>N</i>	integer	The number of colors that define the color space. @ <i>N</i> SHALL be an integer ≥ 1
<i>Name</i> ?	NMTOKEN	Name of the DeviceNSpace.
<i>Separations</i>	NMTOKENS	Ordered list of colorant names that define the DeviceN color space. The number of tokens SHALL match the value of @ <i>N</i> . Additional details of the colorants SHOULD be provided in a Color ResourceSet.

8.20 ColorCorrectionParams

This Resource provides the information needed for an operator to correct colors on some PDL pages or content Elements such as image, graphics or formatted text.

The preferred color adjustment method allows for multi-dimensional adjustments through the use of either an ICC Abstract profile or an ICC DeviceLink profile. The adjustments are not universally colorimetrically calibrated. However, when either of the ICC profile adjustment methods are used, these standard ICC profile formats can be interpreted and applied using generally recognized ICC profile processing techniques. Use of the ICC Abstract profile adjustment will cause the adjustment to be applied in ICC Profile Connection Space, after each source profile is applied, in sequence before final target color conversion. Use of the ICC DeviceLink profile adjustment will cause the adjustment to be applied in final target Device space, after the final target color conversion.

In addition to color adjustment using an ICC profile, the @*AdjustXXX* Attributes each provide a direct color adjustment applied to the interpretation of the PDL data at an implementation dependent point in the processing after each source profile is applied (if source-to-destination color conversion is needed). The L*a*b* values range from -100 to +100 to indicate the minimum and maximum of the range that the system supports. A "0" value means no adjustment. The color adjustment Attributes differ from the Tone Reproduction Curve (TRC) Attributes that can be applied later in the processing path in two key ways. First, the @*AdjustXXX* use, even when included in the Job, will vary as a function of Job content. Second, the data values associated with the @*AdjustXXX* Attributes are arbitrary, and their interpretation will be printer dependent. For details about these Attributes, see Appendix E, "Color Adjustment Attribute Description and Usage" on page 1187.

Resource Properties

Resource referenced by: —

Intent Pairing: **ColorIntent**

Input of Processes: **ColorCorrection**

Output of Processes: —

Table 8-30: ColorCorrectionParams Resource

Name	Data Type	Description
ColorCorrectionOp *	element	List of ColorCorrectionOp Subelements. ColorCorrectionOp SHOULD contain the complete set of parameters for a given color correction operation. Otherwise the results are implementation dependent.
FileSpec (FinalTargetDevice) ?	element	A FileSpec Resource pointing to an ICC profile that describes the characterization of the final output target Device.

8.21 ColorSpaceConversionParams

This set of parameters defines the rules for a **ColorSpaceConversion** Process, the Elements of which define the set of operations to be performed. Information inside the **ColorSpaceConversionOp** Elements defines the operation and identifies the color spaces and types of objects to operate on. Other Attributes define the color management system to use, as well as the working color space and the final target Device.

Resource Properties

Resource referenced by:

Intent Pairing: — **ColorIntent**, **ContentCheckIntent**

Input of Processes: **ColorSpaceConversion**

Output of Processes: —

Table 8-31: ColorSpaceConversionParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>ICCProfileUsage</i> ?	enumeration	<p>This Attribute specifies where to obtain either the destination profile or Device Link transform (see Section 1 on page 365 in Section 6.4.4, “ColorSpaceConversion”) to be applied when converting object colors for the current iteration of the ColorSpaceConversion Process. <i>@ICCProfileUsage</i> provides an order precedence.</p> <p>Note: Use of a final target device profile provides a profiled destination to be used when converting a source object through PCS (Profiled Connection Space) to that profiled destination, and a Device Link transform specifies a conversion directly of the source object from the source space directly to the destination.</p> <p>Allowed values are:</p> <p><i>UsePDL</i> –</p> <ol style="list-style-type: none"> 1 Use the embedded profile. 2 Use the Device Link transform specified in a ColorSpaceConversionOp/FileSpec (<i>DeviceLinkProfile</i>). 3 Use the profile specified in ColorSpaceConversionParams/FileSpec(FinalTargetDevice). 4 Use the system specified profile or Device Link transform. <p><i>UseSupplied</i> –</p> <ol style="list-style-type: none"> 5 Use the Device Link transform specified in a ColorSpaceConversionOp/FileSpec (<i>DeviceLinkProfile</i>). 6 Use the profile specified in ColorSpaceConversionParams/FileSpec(FinalTargetDevice). 7 Use the system specified profile or Device Link transform.

Table 8-31: ColorSpaceConversionParams Resource (Sheet 2 of 2)

Name	Data Type	Description
ColorSpaceConversionOp *	element	<p>List of ColorSpaceConversionOp Elements, each of which identifies a type of object, defines the source color space for that type of object, and specifies the behavior of the conversion operation for that type of object. The XML order of ColorSpaceConversionOp Elements is significant, and when multiple elements apply to the same object, they are applied in that XML order.</p> <p>A ColorSpaceConversionOp can modify the characteristics of an object such that its selection criteria is also modified. Thus, if two ColorSpaceConversionOp Elements select the same set of objects, and the first Element changes the object in such a way that the object would no longer be selected by the second Element, then the second ColorSpaceConversionOp SHALL NOT be applied to that object.</p> <p>ColorSpaceConversionOp SHOULD contain the complete set of parameters for a given color space conversion operation. Otherwise the results are implementation dependent.</p> <p>A ColorSpaceConversionOp process included as part of a RIPping combined process shall include an implied Convert operation as its last operation (causing all other unconverted color spaces to be converted according to the RIP's PDL)</p>
FileSpec (<i>FinalTargetDevice</i>) ?	element	A FileSpec Resource pointing to an ICC profile that describes the characterization of the final output target Device.

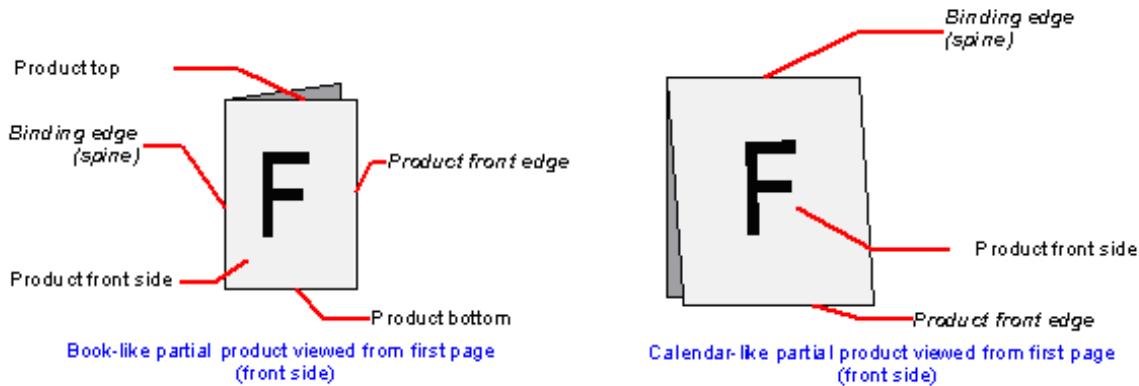
8.22 Component

Component is used to describe the various versions of semi-finished goods in the press and postpress area, such as a pile of folded Sheets that have been collected and are then be joined and trimmed. Nearly every postpress Process has a Resource as an input as well as an output. Typically the first components in the Process chain are some printed Sheets or ribbons, while the last component is a book or a brochure.

Glossary – Component

The descriptions of -specific Attributes use some terms whose meaning depends on the culture in which they are used. For example, different cultures mean different things when they refer to the “front” side of a magazine. Other terms (e.g., binding) are defined by the production process and, therefore, do not depend on the culture.

Whenever possible, this specification endeavors to use culturally independent terms. In cases where this is not possible, Western style (left-to-right writing) is assumed. Please note that these terms might have a different meaning in other cultures (i.e., those writing from right to left).

Figure 8-15: Component – terms and definitions

The table below describes the terms used to define the components.

Table 8-32: Glossary – Component

Term	Definition
Binding edge	The edge on which the (partial) product is glued or stitched. This edge is also often called <i>working edge</i> or <i>spine</i> .
Product front edge	The side, where you open the (partial) product. This edge is opposite to the binding edge.
Registered edge	A side on which a collection of Sheets or partial products is aligned during a production step. All production steps require two registered edges, which SHALL NOT be opposite to each other. The two registered edges define the coordinate system used within the production step. When there is a binding edge, this is one of the registered edges.

Resource Properties

Resource referenced by: *Bundle/BundleItem, DigitalPrintingParams, FeedingParams/Feeder, FeedingParams/CollatingItem*

Input of Processes: *Product Intent, ConventionalPrinting, DigitalPrinting, Varnishing, BlockPreparation, BoxFolding, BoxPacking, Bundling, CaseMaking, CasingIn, Collecting, CoverApplication, Creasing, Cutting, Embossing, EndSheetGluing, Feeding, Folding, Gathering, Gluing, HeadBandApplication, HoleMaking, Inserting, Jacketing, Labeling, Laminating, LooseBinding, Palletizing, Perforating, ShapeCutting, Shrinking, SpinePreparation, SpineTaping, Stacking, StaticBlocking, Stitching, Strapping, ThreadSealing, ThreadSewing, Trimming, WebInlineFinishing, Winding, Wrapping*

Output of Processes: *Product Intent, ConventionalPrinting, DigitalPrinting, Varnishing, BlockPreparation, BoxFolding, BoxPacking, Bundling, CaseMaking, CasingIn, Collecting, CoverApplication, Creasing, Cutting, Embossing, EndSheetGluing, Feeding, Folding, Gathering, Gluing, HeadBandApplication, HoleMaking, Inserting, Jacketing, Labeling, Laminating, LooseBinding, Palletizing, Perforating, ShapeCutting, Shrinking, SpinePreparation, SpineTaping, Stacking, StaticBlocking, Stitching, Strapping, ThreadSealing, ThreadSewing, Trimming, WebInlineFinishing, Winding, Wrapping*

Table 8-33: Component Resource (Sheet 1 of 3)

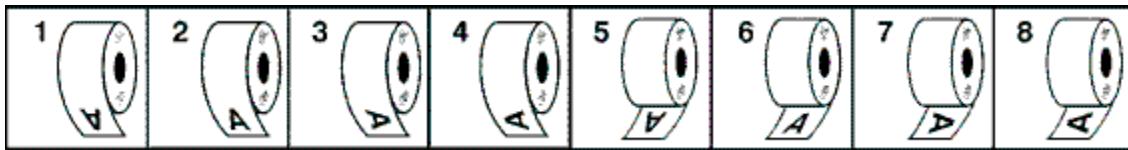
Name	Data Type	Description
<i>AssemblyRef?</i>	IDREF	Specifies the assembly of the . In case of a newspaper-Web Press, the output MAY already be built up of several “booklets”. <i>@AssemblyIDs</i> additionally specifies which <i>AssemblySection</i> Elements of the Assembly belong to this .
<i>Automation?</i>	enumeration	<p>Identifies dynamic and static components.</p> <p>When a is referenced from a Binding process, <i>@Automation</i> modifies the scope of the to be bound. If <i>@Automation="Static"</i>, the individual Elements to be bound are one instance of the referenced . If <i>@Automation="Dynamic"</i>, the individual Elements to be bound are identified by of the referenced Partition. This may either be marked by the availability of all child partitions of the referenced Partition or by the number of Surfaces matching the value of <i>@SurfaceCount</i> specified in the IdentificationField or Pipe XJMF Messages, respectively. The Structure of <i>@PartIDKey</i> generation for automated imposition is defined in detail in: Section 6.4.18.3, Execution Model for Automated Imposition. This structure SHALL be retained in the description.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Static</i> – The is static and completely qualified. <i>Dynamic</i> – The is a template. If <i>@PipeID</i> is also present, Details are specified in XJMF Pipe messages. See Section 11.3.4.1, Dynamic Pipes. If an IdentificationField/ MetadataMap Element is present, the details are controlled by the barcode that is represented by IdentificationField/MetadataMap.
<i>CartonTopFlaps?</i>	XYPair	Size (F1,F2) (See Figure 8-13, “Box packing,” on page 551) of the two top flaps of a carton for shipping. SHALL NOT be specified unless <i>@ProductType = "Carton"</i> .
<i>Columns?</i>	integer	Number of columns of images that are placed on a finished roll, such as by the Winding Process. This value is typically used to describe rolls with multiple columns of printed labels.
<i>ComponentType</i>	enumeration	<p>Specifies the category of the component.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Block</i> – Folded or stacked product (e.g., book block). <i>Other</i> – The describes a sample that has not been produced in this Job. Examples are perfume samples, CDs or toys that are inserted into a printed product. <i>Ribbon</i> – The is a ribbon on a Web Press. <i>Sheet</i> – Single layer (Sheet) of paper. <i>Web</i> – The is a Web on a Web Press. <i>Proof</i> – The is a proof (e.g., a press proof or output from a digital press). <p>Further details of the component are specified in <i>@ProductType</i>.</p>

Table 8-33: Component Resource (Sheet 2 of 3)

Name	Data Type	Description
<i>Dimensions</i> ?	shape	The dimensions of the component. These dimensions MAY differ from the original size of the original product. For example, the dimensions of a folded Sheet MAY be unequal to the dimensions of the Sheet before it was folded. The dimension is always the bounding box around the . If not specified, a portrait orientation (Y > X) is assumed Note: It is crucial for enabling postpress to specify <i>@Dimensions</i> unless they really are unknown.
<i>ContentDataRefs</i> ?	IDREFS	Specification of content metadata for pages described by this .
<i>LayoutRef</i> ?	IDREF	Specifies the original Layout of the source Sheet of the if it contains (@ComponentType, "Sheet") or contains (@ComponentType, "Block"). The original Sheet is the Layout Partition Element where @SourceSheet matches the Layout/@SheetName used in this .
<i>MaxHeat</i> ?	double	Maximum temperature the can resist (in degrees centigrade). The default setting is to impose no restriction in terms of heat (e.g., fusers in electrophotographic Process or shrink wrapping).
<i>MediaRef</i> ?	IDREF	Media for the component. The coordinate system of Media coincides with the coordinate system of the component.
<i>Overfold</i> ?	double	Expansion of the overfold of a . This Attribute is needed for the Inserting or other postpress Processes.
<i>OverfoldSide</i> ?	enumeration	Specifies the longer side of a folded component. Allowed values are: <i>Front</i> <i>Back</i>
<i>ProductType</i> ?	NMTOKEN	Type of product that this component specifies. Values include those from: Table 7-2, “ProductType Attribute Values” on page 440
<i>ProductTypeDetails</i> ?	string	Additional details of the product: If @ProductType = "BlankBox" or @ProductType = "FlatBox", @ProductTypeDetails specifies a box type (e.g., [ECMA], [FEFCO] or company internal box type standard). Values include: <i>NewspaperNormal</i> – Standard newspaper. <i>NewspaperMixed</i> – multiple Resources of a newspaper are produced in parallel. <i>NewspaperCombi</i> – Resources are collected to one in an inline production chain after press.
<i>ReaderPageCount</i> ?	integer	Total amount of individual Reader Pages that this contains. Count of -1 means “unknown.” If not specified, the value is unknown.
<i>SheetPart</i> ?	rectangle	Only used if contains (@ComponentType, "Block") and Layout is present. Position of the block on the Layout in @SurfaceContentsBox coordinates used in this .

Table 8-33: Component Resource (Sheet 3 of 3)

Name	Data Type	Description
<i>SpineThickness</i> ?	double	Thickness
<i>SurfaceCount</i> ?	integer	Total amount of individual surfaces that this contains. Note: a sheet always has 2 Surfaces regardless of the number of images or reader pages. In case of homogeneous Elements, <i>@SurfaceCount</i> refers to surfaces with a size of / <i>@Dimensions</i>
<i>WindingResult</i> ?	integer	Orientation of the finished product on the Roll. For an image, see Figure 8-20, “Orientation of the Finished Product on the Roll,” on page 593. The integer in the figure corresponds to value specified by http://www.finat.org . Note: the orientation and number of windings in a Winding Process are modified based on the value of <i>@WindingResult</i> .
Bundle ?	element	Description of a Bundle of Resources if the represents multiple individual items. If no Bundle is present, the represents an individual item. Note that it is essential to keep a reference of the child Resources that comprise a , as this information is useful to postpress Processes.
Disjointing ?	element	A stack of components can be processed using physical separators. This is useful in operations such as feeding.

Figure 8-16: Orientation of the Finished Product on the Roll

8.23 Contact

Element describing a contact to a person or address. The *@ExternalID* attribute in the parent Resource element SHALL be unique within the company.

Resource Properties

Resource referenced by: **ApprovalParams**/**ApprovalPerson**, **ApprovalDetails** , **Content**/
ContentMetadata, **CustomerInfo**, **DeliveryParams**,

Input of Processes: —

Output of Processes: —

Table 8-34: Contact Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>ContactTypeDetails</i> ?	string	<p><i>@ContactTypeDetails</i> specifies the details of the Contact's role or roles.</p> <p>If <i>@ContactTypes</i> contains "<i>Employee</i>", then <i>@ContactTypeDetails</i> defines the list of roles that the employee fills.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Apprentice</i> – Employee that is in training (“Auszubildender” / “Auszubildende” / "Stift" in German). <i>Assistant</i> – Assistant operator. <i>Craftsman</i> – Trained employee (“Geselle” / “Facharbeiter” in German). <i>CSR</i> – Customer Service Representative <i>Manager</i> – Manager. <i>Master</i> – Highly trained employee (“Meister” in German). <i>Operator</i> – Operator. <i>ShiftLeader</i> – The leader of the shift. <i>StandBy</i> – Employee who is allocated to a specific task on demand.

Table 8-34: Contact Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>ContactTypes</i>	NMTOKENS	<p>One or more roles of the contact.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Accounting</i> – Address of where to send to the bill. <i>Administrator</i> – Person to contact for queries concerning the execution of the Job. <i>Agency</i> – The contact is an employee of an Agency. <i>Approver</i> – Person who approves this Job. <i>ArtReturn</i> – Return delivery or pickup address for artwork of this Job. <i>Author</i> – <i>Billing</i> – Contact information that refers to a payment method (e.g., credit card). <i>Customer</i> – The end customer. <i>Delivery</i> – The delivery address for all products of this Job. <i>DeliveryCharge</i> – The Contact is charged for delivery of this Job. <i>Designer</i> – <i>Editor</i> – <i>Employee</i> – Employee who works for the printer that is processing this job. <i>Illustrator</i> – <i>Owner</i> – The owner of a Resource. <i>Photographer</i> – <i>Pickup</i> – The pickup address for all products of this Job. <i>Sender</i> – The source address of the delivery. <i>Supplier</i> – Address of a supplier of needed goods. <i>SurplusReturn</i> – Return delivery or pickup address for surplus products of this Job. <i>TelephoneSanitizer</i> –
<i>CostCenterID</i> ?	NMTOKEN	Identifier of the cost center that this Contact belongs to if <i>@ContactTypes</i> contains "Employee"
<i>UserID</i> ?	string	User ID of user, as specified when logging into the operating system or into the submitting application.
<i>Address</i> ?	element	Element describing the address.
<i>ComChannel</i> *	element	Communication channels to the contact. These Elements define communication channels that MAY be assigned to multiple Persons, for instance the communication channel of a reception area.
<i>Company</i> ?	element	Company that this Contact is associated with.
<i>Person</i> ?	element	Name of the contact person.

8.23.1 Element: ComChannel

A communication channel to a person or company such as an email address, phone number or fax number.

Table 8-35: ComChannel Element

Name	Data Type	Description
<i>ChannelType</i>	NMTOKEN	<p>Type of the communication channel.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>ComputerName</i> <i>Email</i> – Email address. <i>Fax</i> – Fax machine. <i>JMF</i> – XJMF messaging channel. <i>Mobile</i> – Mobile phone. <i>Phone</i> – Telephone number. This SHOULD be restricted to land line phones. <i>WWW</i> – WWW home page or form. <p><i>PrivateDirectory</i> – Account of a registered customer of a certain service. (The list of the registered accounts is maintained by the service vendor). The <i>ChannelTypeDetails</i> Attribute has the name of the private directory service vendor.</p> <p><i>InstantMessaging</i> – IM service address. The <i>ChannelTypeDetails</i> Attribute has the name of the IM service vendor</p>
<i>ChannelUsage?</i>	NMTOKENS	<p>Communication channel usage.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Business</i> – Business purpose usage (e.g., office phone number, fax). <i>Private</i> – Private purpose usage (e.g., private phone number, fax, Email). <i>DayTime</i> – Office hours in the time zone of the recipient. <i>NightTime</i> – Out-of-office hours in the time zone of the recipient. <i>WeekEnd</i> – Out-of-office hours in the time zone of the recipient.
<i>Locator</i>	string	<p>Locator of this type of channel in a form, such as a phone number, a URL or an Email address. If a URL is defined for the <i>@ChannelType</i>, it is RECOMMENDED to use the URL syntax specified in [RFC2368] for “mailto” URLs, [RFC3966] for “tel” URLs and [RFC3986] for URLs in general, as follows:</p> <p>Values include:</p> <ul style="list-style-type: none"> “mailto:<i>a@b.com</i>” – instead of “<i>a@b.com</i>” if <i>@ChannelType</i> = “Email”, “tel:+49-69-92058800” – if <i>@ChannelType</i> = “Phone” and “tel:+49.6151.155.299” – if <i>@ChannelType</i> = “Fax”.

— Attribute: **ChannelTypeDetails**

Table 8-36: ChannelTypeDetails Attribute – predefined values for certain ChannelType values (Sheet 1 of 2)

ChannelType value	ChannelTypeDetails value	Description
“Phone”	“Secure”	Secure phone line.
	“ISDN”	ISDN line telephone number.

Table 8-36: ChannelTypeDetails Attribute—predefined values for certain ChannelType values (Sheet 2 of 2)

ChannelType value	ChannelTypeDetails value	Description
"urn:nii"	"#form"	Upload form.
"#target"		Upload target URL.

8.23.2 Element: Company

Specifies contacts to a company including detailed information about contact persons and addresses. This structure can be used in many situations where addresses or contact persons are needed. Examples of contacts are customer, supplier, company and addressees. The structure is derived from the vCard format. It comprises the organization name and organizational units (ORG) of the organizational properties defined in the vCard format. The corresponding XML types of the vCard are quoted in the table. The *@ExternalID* attribute SHALL be unique across all companies.

Table 8-37: Company Element

Name	Data Type	Description
<i>CompanyID</i> ?	ENMTOKEN	An ID of the Company. The <i>@CompanyID</i> attribute for each specified company SHALL be unique within the company (i.e., the company's database).
<i>OrganizationName</i>	string	Name of the organization or company (vCard: ORG:orgnam (e.g., ABC, Inc.)).
<i>OrganizationalUnit</i> *	text element	Describes the organizational unit (vCard: ORG:orgunit. For example, if two Elements are present: 1. "North American Division" and 2. "Marketing").

8.23.3 Element: Person

This Element provides detailed information about a person. It also has the ability to specify different communication channels to this person. Use *@ExternalID* when a unique identifier for the Person is required. The structure of the Element is derived from the vCard format. It contains all of the same name subtypes (N:) of the identification and the title of the organizational properties. The corresponding XML types of the vCard are quoted in the description field of the table below.

Table 8-38: Person Element

Name	Data Type	Description
<i>AdditionalNames</i> ?	string	Additional names of the contact person (vCard: N:other).
<i>FamilyName</i> ?	string	The family name of the contact person (vCard: N:family).
<i>FirstName</i> ?	string	The first name of the contact person (vCard: N:given).
<i>JobTitle</i> ?	string	Job function of the person in the company or organization (vCard: title).
<i>Languages</i> ?	languages	List of languages related to the person, ordered by decreasing preference
<i>NamePrefix</i> ?	string	Prefix of the name, can include title (vCard: N:prefix).
<i>NameSuffix</i> ?	string	Suffix of the name (vCard: N:suffix).
<i>PhoneticFirstName</i> ?	string	Alternative spelling of a first name. Used (e.g., for pronunciation of Kanji (Japanese) names). See http://en.wikipedia.org/wiki/VCard .
<i>PhoneticLastName</i> ?	string	Alternative spelling of a last name. Used (e.g., for pronunciation of Kanji (Japanese) names). See http://en.wikipedia.org/wiki/VCard .

8.24 Content

Resource Properties

Resource referenced by: **RunList**

Input of Processes: —

Output of Processes: —

Content defines the additional metadata of individual graphic elements.

Table 8-39: Content Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>BinderySignatureIDs?</i>	NMTOKENS	IDs of the Assembly Elements, AssemblySection Elements or BinderySignature / @AssemblyID that this finished page belongs to
<i>ContentStatus?</i>	NMTOKENS	Status of a single Content Element. Values include those defined in the column "ContentStatus" of Table C-20, "MessageEvents and MilestoneType Values" on page 1178.
<i>ContentType?</i>	NMTOKEN	Type of content. Values include those from: Table 8-58, "ContentType Attribute Values" on page 603.
<i>HasBleeds?</i>	boolean	If "true", the Content has bleeds.
<i>ImageCompressionParamsRef?</i>	IDREF	Specification of the image compression properties.
<i>IsBlank?</i>	boolean	If "true", the Content has no content marks and is blank.
<i>IsTrapped?</i>	boolean	If "true", the Content has been trapped.
<i>PageLabel?</i>	string	Complete identification of the finished page as it is displayed on the finished page, For instance "1", "iv" or "C-1". Note that this MAY be different from the position of the page in the finished document.
<i>ScreeningParamsRef?</i>	IDREF	Specification of the screening properties of the Content .
<i>Separations?</i>	NMTOKENS	List of separation names defined in the Content .
<i>SourceBleedBox?</i>	rectangle	A rectangle that describes the bleed area of the content to be included. This rectangle is expressed in the source coordinate system of the object.
<i>SourceClipBox?</i>	rectangle	A rectangle that defines the region of the finished content to be included. This rectangle is expressed in the source coordinate system of the object.
<i>SourceTrimBox?</i>	rectangle	A rectangle that describes the intended trimmed size of the finished content to be included. This rectangle is expressed in the source coordinate system of the object.
<i>BarcodeProductionParams?</i>	element	Description of the specific s for barcode production.
<i>ContentMetadata?</i>	element	Container for document related metadata such as ISBN, Author etc.

Table 8-39: Content Resource (Sheet 2 of 2)

Name	Data Type	Description
FileSpec *	element	Reference to dependent references such as fonts, external images, etc. FileSpec/@ResourceUsage SHOULD have one of the following values: <i>Font</i> – The file references a font. <i>Image</i> – The file references Image data. <i>PDL</i> – The file references page definition language data such as PDF.
OCGControl *	element	@OCGControl provides a list of the OCGs (layers) that SHALL be included or excluded. Any OCGs (layers) not listed in an OCGControl Element SHALL follow the rules defined by the underlying PDL.
PositionObj *	element	Definition of the size and positioning of any child Content elements

— Attribute: ContentType**Table 8-40: ContentType Attribute Values**

Value	Description
<i>Ad</i>	The content represents a single ad
<i>Article</i>	The content represents a single article. Including headers, text bodies, photos etc.
<i>Barcode</i>	A barcode.
<i>ClassifiedAd</i>	Specifies a classified ad.
<i>ClassifiedsPageElement</i>	Specifies a grouping page element dealing with content of classified ads
<i>Composed</i>	Combination of elements that define an element that is not bound to a document page.
<i>Editorial</i>	Defines this Element to contain editorial matter (e.g., text, photos etc.).
<i>EditorialPageElement</i>	Specifies a grouping page element dealing with content of the editorial department
<i>Graphic</i>	Line art.
<i>IdentificationField</i>	A general identification field excluding bar codes.
<i>Image</i>	Bitmap image.
<i>Page</i>	Representation of one document page.
<i>PageHeader</i>	For instance a newspaper title shown on the front page or on each single page. Usually, these headers contain information like page number, editorial desk and the date.
<i>ROPAd</i>	Specifies this Element as an ROP ad. An ROP ad is an ad which is placed by the planner. Generally speaking, in a newspaper environment these include color ads, ads with placement requests in the editorial section and large ads.
<i>Surface</i>	Representation of an imposed surface.
<i>Text</i>	Formatted or unformatted text.

8.24.1 Element: BarcodeProductionParams

BarcodeProductionParams describes of the specific parameters for barcode production.

Table 8-41: BarcodeProductionParams Element

Name	Data Type	Description
IdentificationFieldRef?	IDREF	Description of the barcode metadata.
BarcodeReproParams?	element	Description of the formatting and reproduction parameters for barcode production.

8.24.2 Element: ContentMetadata

ContentMetadata is a container for metadata pertaining to this **Content** Resource.

Table 8-42: ContentMetadata Element

Name	Data Type	Description
ContactRefs?	IDREFS	The person who is responsible for this content.
ISBN10?	NMTOKEN	The 10 digit ISBN (see ref ISBN)
ISBN13?	NMTOKEN	The 13 digit ISBN (see ref ISBN)
Title?	NMTOKEN	The title of the content
Comment?	element	If required, an abstract should be specified in Comment[@Name = "Abstract"].
GeneralID*	element	Additional metadata MAY be defined by adding GeneralID elements.

8.24.3 Element: PositionObj

PositionObj describes the size and position of the **Content**.

Table 8-43: PositionObj Element (Sheet 1 of 2)

Name	Data Type	Description
Anchor?	Anchor	@Anchor specifies the origin (0,0) of the coordinate system in the unrotated Content .
CTM?	matrix	Transformation matrix of the origin of Content as specified by @Anchor. Note that this is not necessarily the actual CTM that will position a given Content . The actual CTM SHALL be recalculated based on the values of @Anchor and @Size.
PageRange?	IntegerRange	Reader Page index in the Content referenced by RefAnchor.
PositionPolicy?	enumeration	Specifies the level of freedom when applying the values specified in PositionObj. Allowed values are: <i>Exact</i> – The values SHALL be followed precisely. <i>Free</i> – The values are used as guidance and MAY be modified by the designer.
RelativeSize?	XYPair	Specifies the size of the unrotated and unscaled object, relative to the parent specified in RefAnchor.

Table 8-43: PositionObj Element (Sheet 2 of 2)

Name	Data Type	Description
<i>RotationPolicy</i> ?	enumeration	<p>Specifies the level of freedom when applying the values specified in PositionObj.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Exact</i> – The values SHALL be followed precisely. <i>Free</i> – The values are used as guidance and MAY be modified by the designer.
<i>Size</i> ?	XYPair	Specifies the size of the unrotated and unscaled object, in points.
<i>SizePolicy</i> ?	enumeration	<p>Specifies the level of freedom when applying the values specified in PositionObj.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Exact</i> – The values SHALL be followed precisely. <i>Free</i> – The values are used as guidance and MAY be modified by the designer.
<i>RefAnchor</i> ?	element	<p>Reference to a Content that this Content is positioned relative to.</p> <p>If RefAnchor is not specified, PositionObj refers to the lower left of the first page specified in page Range.</p>

Example 8-10: ContentList

TBD 2.x Example.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1"
      ID="n071030_02242300_000002" JobPartID="n071030_02242300_000002"
      Status="Waiting" Type="Approval" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="Approval">
    <!--Generated by the CIP4 Java open source JDF Library version : CIP4 JDF
        Writer Java 1.3 BLD 47-->
    <AuditPool>
        <Created AgentName="CIP4 JDF Writer Java" AgentVersion="1.3 BLD 47"
                  ID="a071030_02242331_000003"TimeStamp="2007-10-30T14:24:23+01:00"/>
    </AuditPool>
    <ResourcePool>
        <ApprovalParams Class="Parameter" ID="ID345" Status="Available"/>
        <ApprovalSuccess Class="Parameter" ID="ID346" Status="Unavailable"/>
        <RunList Class="Parameter" ID="r071030_02242378_000004"
                  Status="Available">
            <PageListRef rRef="PageList"/>
        </RunList>
        <PageList Class="Parameter" ID="PageList" Status="Available">
            <ContentListRef rRef="ContentList"/>
        </PageList>
        <ContentList Class="Parameter" ID="ContentList" Status="Available">
            <ContentData>
                <ContentMetadata ISBN10="0123456789" Title="book thing">
                    <Comment ID="c071030_022423109_000005" Name="Abstract">
                        Abstract of the book in english
                    </Comment>
                    <Contact Class="Implementation" ContactTypes="Editor">
                        <Person Class="Parameter" DescriptiveName="authorName"
                               FamilyName="authorName"/>
                    </Contact>
                </ContentMetadata>
            </ContentData>
        </ContentList>
    </ResourcePool>
</JDF>
```

```

        </Contact>
    </ContentMetadata>
</ContentData>
<ContentData>
    <ContentMetadata Title="Chapter 1">
        <Contact Class="Implementation" ContactTypes="Customer">
            <Person Class="Parameter" DescriptiveName="authorName1"
                FamilyName="authorName1"/>
        </Contact>
    </ContentMetadata>
</ContentData>
<ContentData>
    <ContentMetadata Title="Chapter 2">
        <Contact Class="Implementation" ContactTypes="Customer">
            <Person Class="Parameter" DescriptiveName="authorName2"
                FamilyName="authorName2"/>
        </Contact>
    </ContentMetadata>
</ContentData>
<ContentData>
    <ContentMetadata Title="Chapter 3">
        <Contact Class="Implementation" ContactTypes="Customer" >
            <Person Class="Parameter" DescriptiveName="authorName3"
                FamilyName="authorName3"/>
        </Contact>
    </ContentMetadata>
</ContentData>
</ContentList>
</ResourcePool>
<ResourceLinkPool>
    <RunListLink Usage="Input" rRef="r071030_02242378_000004"/>
    <ApprovalParamsLink Usage="Input" rRef="ID345"/>
    <ApprovalSuccessLink Usage="Output" rRef="ID346"/>
</ResourceLinkPool>
</JDF>

```

Example 8-11: ContentList: Extended with ISBN, Author, etc.

Example of **Content** with ISBN, Author, etc.

TBD 2.x Example.

```

<!-- Information about the input (file, author) -->
<RunList ID="NodeIDRunList" Status="Available" Class="Parameter" >
    <LayoutElementRef rRef="NodeIDLE" />
    <PageList>
        <ContentList>
            <ContentData>
                <!-- String for title -->
                <new:DocumentInfo Title="This is the title of the book"
                    ISBN="0123456789" xmlns:new="new schema URI">
                    <!-- Multi-lines string for Abstract -->
                    <new:DocumentAbstract>
                        This is the abstract of the book
                        It has several lines...
                    </new:DocumentAbstract>
                    <!-- List of authors. Using a PersonRef allows reusing the same
                        Person element -->
                    <new:Author Subject="Preface">

```

```

        <PersonRef rRef="AuthorID1" />
    </new:Author>
    <new:Author Subject="Content">
        <new:PersonRef rRef="AuthorID2" />
        <new:PersonRef rRef="AuthorID3" />
    </new:Author>
    </new:DocumentInfo>
</ContentData>
</ContentList>
</PageList>
</RunList>
<LayoutElement ID="NodeIDLE" Status="Available" Class="Parameter" >
    <FileSpec URL="file:///hotfolder/files/Document2747.pdf"
       MimeType="application/pdf" UserFileName="JDF1.3.pdf" />
</LayoutElement>

<Person ID="AuthorID1" Class="Parameter" Status="Available" FirstName="James"
    FamilyName="Smith" JobTitle="Author" />
<Person ID="AuthorID2" Class="Parameter" Status="Available" FirstName="John"
    FamilyName="Smith" JobTitle="Author" />
<Person ID="AuthorID3" Class="Parameter" Status="Available" FirstName="William"
    FamilyName="Smith" JobTitle="Author" />
<!-- Media: A3 white paper coated on both sides, 70 gr/m2 --&gt;
&lt;Media ID="MediaID" Class="Consumable" Status="Available" Weight="70"
    Dimension="1190 842" MediaType="Paper" MediaColorName="White"
    FrontCoatings="Coated" BackCoatings="Coated" /&gt;
<!-- Media: A4 yellow paper for Banner Page --&gt;
&lt;Media ID="MediaID2" Class="Consumable" Status="Available" Weight="70"
    Dimension="595 842" MediaType="Paper" MediaColorName="Yellow" /&gt;
<!-- Booklet layout + banner page with ISBN and Authors printed on it --&gt;
&lt;LayoutPreparationParams ID="NodeIDLPP" Class="Parameter" Status="Available"
    Sides="TwoSidedFlipY" NumberUp="2 1" BindingEdge="Left"
    PresentationDirection="FoldCatalog" FoldCatalog="F4-1"
    FinishingOrder="GatherFold" PageDistributionScheme="Saddle"&gt;
    &lt;InsertSheet SheetType="JobSheet" SheetUsage="Header" IsWaste="true"
        SheetFormat="Standard" &gt;
        &lt;Layout&gt;
            &lt;MediaRef rRef="MediaID2" /&gt;
            &lt;MarkObject CTM="1 0 0 1 0 0" &gt;
                &lt;JobField ShowList="new:ISBN new:Authors" /&gt;
            &lt;/MarkObject&gt;
        &lt;/Layout&gt;
    &lt;/InsertSheet&gt;
&lt;/LayoutPreparationParams&gt;
</pre>

```

8.25 ConventionalPrintingParams

This Resource defines the Attributes and Elements of the **ConventionalPrinting** Process. The specific parameters of individual printer modules are modeled by using the standard Partitioning methods. These methods are described in Section 3.13.2.2, “Selecting a Partition”

Resource Properties

Resource referenced by: —

Input of Processes: **ConventionalPrinting**

Output of Processes: —

Table 8-44: ConventionalPrintingParams Resource

Name	Data Type	Description
<i>Drying</i> ?	enumeration	The way in which ink is dried after a print run. Allowed values are: <i>UV</i> – Ultraviolet dryer <i>Heatset</i> – Heatset dryer <i>IR</i> – Infrared dryer <i>On</i> – Use the Device default drying unit. <i>Off</i>
<i>FirstSurface</i> ?	enumeration	Printing order of the surfaces on the Sheet. Allowed values are: <i>Front</i> <i>Back</i>
<i>FountainSolution</i> ?	enumeration	State of the fountain solution module in the printing units. Allowed values are: <i>On</i> <i>Off</i>
<i>ModuleDrying</i> ?	enumeration	The way in which ink is dried in individual modules. Allowed values are: <i>UV</i> – Ultraviolet dryer <i>Heatset</i> – Heatset dryer <i>IR</i> – Infrared dryer <i>On</i> – Use the Device default drying unit. <i>Off</i>
<i>PerfectingModule</i> ?	integer	Index of the perfecting module if @WorkStyle = " <i>Perfecting</i> " and multiple perfecting modules are installed.
<i>Powder</i> ?	double	Quantity of powder in%.
<i>SheetLay</i> ?	enumeration	Lay of input media. Reference edge of where paper is placed in a feeder. Allowed values are: <i>Left</i> <i>Right</i> <i>Center</i>
<i>Speed</i> ?	double	Maximum print speed in Sheets/hour (Sheet fed) or revolutions/hour (Web-Fed). Defaults to Device specific full speed.
<i>WorkStyle</i> ?	WorkStyle	The direction in which to turn the press Sheet.
<i>ApprovalParams</i> ?	element	Details of the direct Approval Process, when @DirectProof = "true".

8.26 CoverApplicationParams

CoverApplicationParams define the parameters for applying a cover to a book block.

Resource Properties

Resource referenced by:

Intent Pairing:

BindingIntent

Input of Processes:

CoverApplication

Output of Processes:

—

Table 8-45: CoverApplicationParams Resource

Name	Data Type	Description
GlueLine *	element	Describes where and how to apply glue to the book block.
Score *	element	Describes where and how to score the cover. The sequence of Score Elements specifies the sequence in which the tool is applied.

8.26.1 Element: Score**Table 8-46: Score Element**

Name	Data Type	Description
Offset	double	Position of scoring given in the operation coordinate system.
Side ?	enumeration	Specifies the side from which the scoring tool works. Allowed values are: <i>FromInside</i> <i>FromOutside</i>

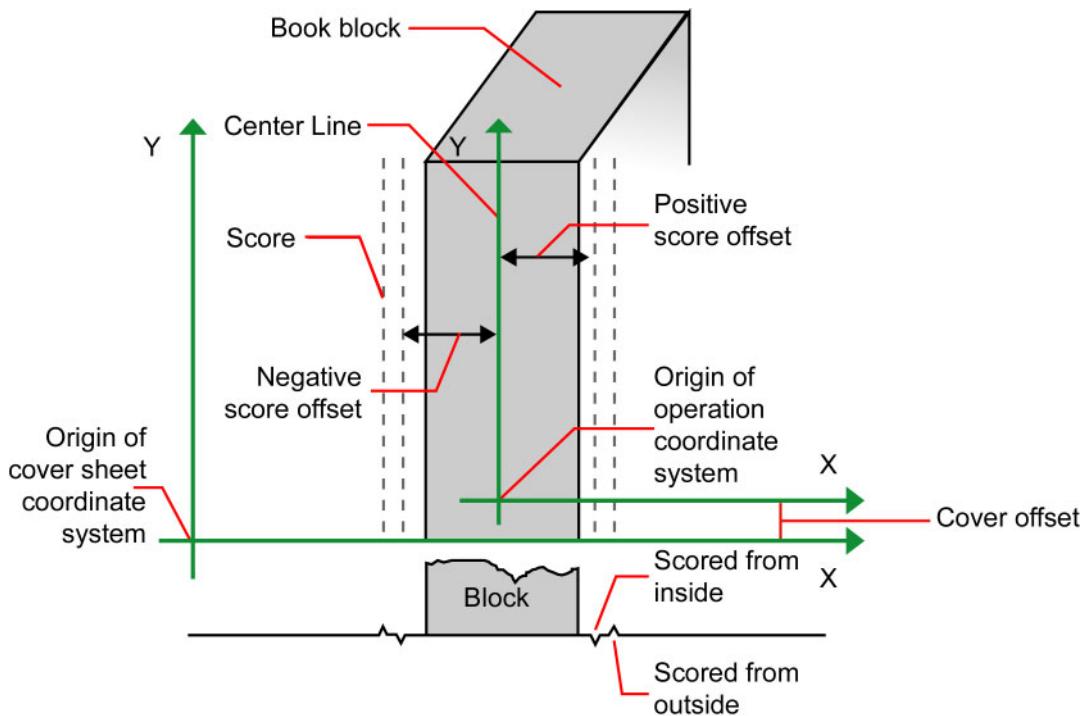
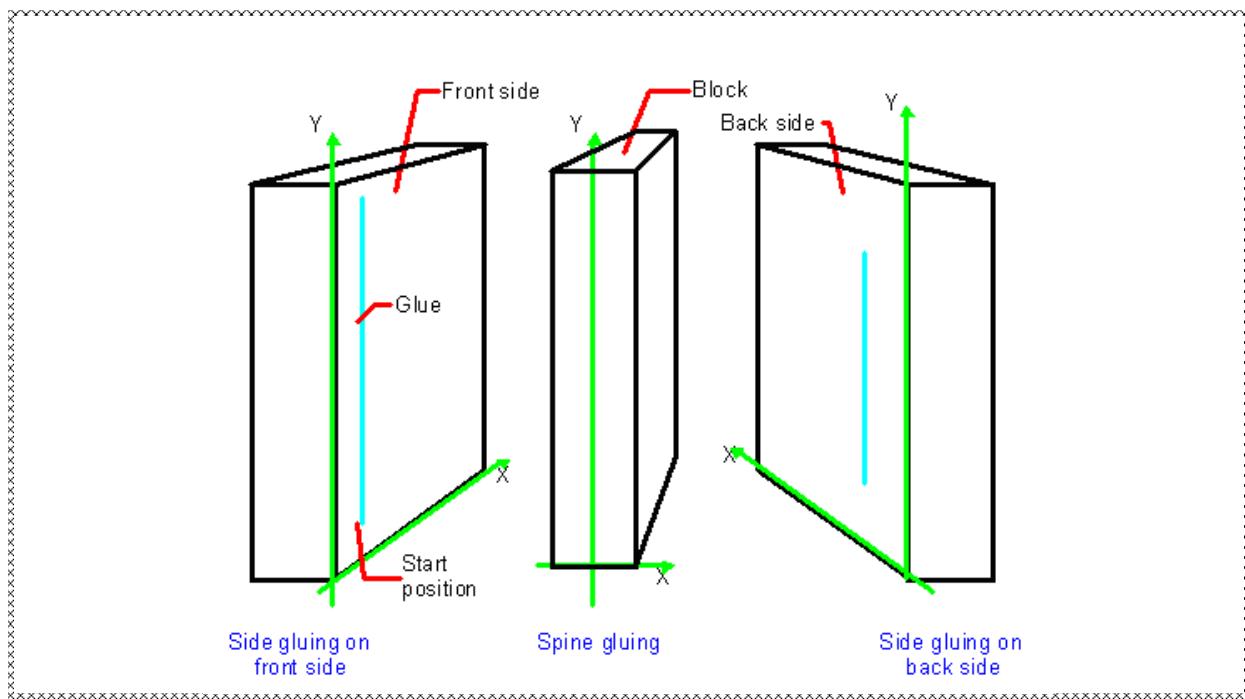
Figure 8-17: Parameters and coordinate system for cover application

Figure 8-18: Parameters and coordinate system for glue application

8.27 CreasingParams

CreasingParams define the parameters for creasing or grooving a Sheet.

Resource Properties

Resource referenced by: —

Intent Pairing: **FoldingIntent**

Input of Processes: **Creasing**

Output of Processes: —

Table 8-47: CreasingParams Resource

Name	Data Type	Description
Crease *	element	Each Crease element defines one crease line.

CustomerInfo The **CustomerInfo** Resource contains information about the customer who orders the Job.

Resource Properties

Resource referenced by: —

Input of Processes: **Any Process**

Output of Processes: —

Table 8-48: CustomerInfo Resource (Sheet 1 of 2)

Name	Data Type	Description
BillingCode ?	string	A code to bill charges incurred while executing the Node.

Table 8-48: CustomerInfo Resource (Sheet 2 of 2)

Name	Data Type	Description
ContactRefs ?	IDREFS	Resource Element describing contacts associated with the customer. There SHOULD be one Contact [contains (@ContactTypes, "Customer")]. Such a Contact specifies the primary customer's name, address etc.
CustomerID ?	NMTOKEN	Customer identification used by the application that created the Job. This is usually the internal customer number of the MIS system that created the Job.
CustomerJobName ?	string	The human readable descriptive name that the customer uses to refer to the Job.
CustomerOrderID ?	string	The internal order number in the system of the customer. This number is usually provided when the order is placed and then referenced on the order confirmation or the bill.
CustomerProjectID ?	string	The internal project id in the system of the customer. This number MAY be provided when the order is placed and then referenced on the order confirmation or the bill.

8.28 Resource Resource CuttingParams

This Resource describes the parameters of a **Cutting** Process that uses either **Cut** elements or **CutBlock** Elements as input. If **CuttingParams** is partitioned by **@BlockName**, **Part/@BlockName** SHALL specify the input **CutBlock**.

Resource Properties

Resource referenced by:

—

Intent Pairing:

FoldingIntent, ShapeCuttingIntent

Input of Processes:

Cutting

Output of Processes:

—

Table 8-49: CuttingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
SheetLay ?	enumeration	Lay of input Component . Allowed values are: <i>Left</i> <i>Right</i> Note: @SheetLay does not modify the coordinate references of the Cutting Process.
Cut *	element	Cut Elements describe an individual cut. Cut Elements SHALL NOT be specified if CutBlock Elements are specified.
CutBlock *	element	These CutBlock elements describe the output cut blocks that SHALL be cut out of the input CutBlock or Component . The CutBlock Elements SHALL NOT be written if Cut Elements are specified.

Table 8-49: CuttingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
FileSpec (CIP3) ?	element	Reference to a CIP3 file that contains cutting instructions in the CIP3 format. The FileSpec/@ResourceUsage = "CIP3".

8.28.1 Element: CutBlock

CutBlock specifies exactly one cut block on a sheet. If **Part/@BlockName** is specified for the ancestor Resource element, then the **CutBlock** SHALL be defined in the coordinate system of the parent **CutBlock** with **CutBlock/@BlockName= Part/@BlockName**. Otherwise the input **Component** SHALL be the parent sheet. Each cut block has its own coordinate system, which has its origin (0,0) at the lower left corner of the **CutBlock** and the same orientation as the parent component or **CutBlock**.

Table 8-50: CutBlock Element

Name	Data Type	Description
BinderySignatureIDs ?	NMTOKENS	The @AssemblyIDs of the Assembly , AssemblySection or BinderySignature which are contained in this CutBlock . If specified, @BinderySignatureIDs SHALL list the BinderySignature/@BinderySignatureID of all BinderySignatures that comprise this CutBlock .
BlockElementType ?	enumeration	Element type. Allowed values are: <i>CutElement</i> – Cutting element. <i>PunchElement</i> – Punching element.
BlockName	NMTOKEN	Name of the block. Used for reference by the CutMark Resource.
BlockType	enumeration	Block type. Allowed values are: <i>CutBlock</i> – Block to be cut. <i>SaveBlock</i> – Protected block, cut only via outer contour. <i>TempBlock</i> – Auxiliary block that is not taken into account during cutting. <i>MarkBlock</i> – Contains no elements, only marks.
Box ?	rectangle	Defines the position and size of the block relative to the parent Component or CutBlock coordinate system.
CutWidth ?	double	Width in points of u-shaped knife, saw blade, etc.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="Bundle" Status="Waiting"
      Type="CylinderLayoutPreparation" JobPartID="ID20" Version="1.4">
  <ResourcePool>
    <CylinderLayoutPreparationParams ID="CL002" Class="Parameter"
        Status="Available" >
      <ProductionPath/>
    </CylinderLayoutPreparationParams>
    <RunList ID="R-002" Class="Parameter" Status="Available" />
    <Device ID="DEV-001" Manufacturer="MAN" ModelName="GEOMAN" Status="Available"
          Class="Implementation" DeviceID="DEV-001">
      <Module ModuleIndex="0" ModuleType="Folder" ModelName="Folder 1">
        <Module ModuleIndex="1" ModuleType="PrintUnit"
              DescriptiveName="PU-1">
```

```

<Module ModuleIndex="2" SubModuleIndex="0"
        ModuleType="PrintModule" DescriptiveName="PM-1"/>
<Module ModuleIndex="3" SubModuleIndex="1"
        ModuleType="PrintModule" DescriptiveName="PM-2"/>
<Module ModuleIndex="4" SubModuleIndex="2"
        ModuleType="PrintModule" DescriptiveName="PM-3"/>
<Module ModuleIndex="5" SubModuleIndex="3"
        ModuleType="PrintModule" DescriptiveName="PM-4"/>
<Module ModuleIndex="6" SubModuleIndex="4"
        ModuleType="PrintModule" DescriptiveName="PM-5"/>
<Module ModuleIndex="7" SubModuleIndex="5"
        ModuleType="PrintModule" DescriptiveName="PM-6"/>
<Module ModuleIndex="8" SubModuleIndex="6"
        ModuleType="PrintModule" DescriptiveName="PM-7"/>
<Module ModuleIndex="9" SubModuleIndex="7"
        ModuleType="PrintModule" DescriptiveName="PM-8"/>
</Module>
</Module>
</Device>
<Layout ID="L-001" Class="Parameter" Status="Available"/>
<CylinderLayout ID="CL-001" Class="Parameter" Status="Available"
    PartIDKeys="WebSetup PlateLayout Separation"
    DeviceID="DEV-001">
    <LayoutRef rRef="L-001"/>
    <CylinderLayout WebSetup="Run-1">
        <CylinderLayout PlateLayout="PL-001">
            <CylinderLayout Separation="Yellow">
                <CylinderPosition DeviceModuleIndex="2" PlatePosition="0 0"
                    PlateType="Exposed" PlateUsage="Original"/>
                <!-- page 1 -->
                <CylinderPosition DeviceModuleIndex="2" PlatePosition="0 1"
                    PlateType="Exposed" PlateUsage="Original"/>
                <!-- page 1 -->
            </CylinderLayout>
        </CylinderLayout>
    </CylinderLayout>
    <CylinderLayout PlateLayout="PL-002">
        <CylinderLayout Separation="Yellow">
            <CylinderPosition DeviceModuleIndex="2" PlatePosition="1 0"
                PlateType="Exposed" PlateUsage="Original"/>
            <!-- page 8 -->
            <CylinderPosition DeviceModuleIndex="2" PlatePosition="1 1"
                PlateType="Exposed" PlateUsage="Original"/>
            <!-- page 8 -->
        </CylinderLayout>
    </CylinderLayout>
    <CylinderLayout PlateLayout="PL-003">
        <CylinderLayout Separation="HKS57">
            <CylinderPosition DeviceModuleIndex="2" PlatePosition="2 0"
                PlateType="Exposed" PlateUsage="Reuse"/>
            <!-- page 3 -->
            <CylinderPosition DeviceModuleIndex="2" PlatePosition="2 1"
                PlateType="Exposed" PlateUsage="Reuse"/>
            <!-- page 3 -->
        </CylinderLayout>
    </CylinderLayout>
    <CylinderPosition DeviceModuleIndex="2" PlatePosition="3 0"
        PlateType="Dummy" PlateUsage="Reuse"/>
    <CylinderPosition DeviceModuleIndex="2" PlatePosition="3 1"

```

```

        PlateType="Dummy" PlateUsage="Reuse"/>
    </CylinderLayout>
</CylinderLayout>
</ResourcePool>
<ResourceLinkPool>
    <DeviceLink Usage="Input" rRef="DEV-001"/>
    <LayoutLink Usage="Input" rRef="L-001"/>
    <RunListLink Usage="Input" rRef="R-002"/>
    <CylinderLayoutPreparationParamsLink Usage="Input" rRef="CL002"/>
        <CylinderLayoutLink Usage="Output" rRef="CL-001"/>
    </ResourceLinkPool>
</JDF>

<CylinderLayout ID="CL-001" Class="Parameter" Status="Available"
    PartIDKeys="WebSetup PlateLayout Separation"
    DeviceID="DEV-001">
    <!-- ... -->
    <!-- PlatePosition (XYPairRangeList) -->
    <CylinderPosition DeviceModuleIndex="2" PlatePosition="0 0 ~ 1 0"
        PlateType="Exposed" PlateUsage="Original"/>
    <!-- ... -->
</CylinderLayout>

```

8.29 DeliveryParams

DeliveryParams provides information needed by a **Delivery** Process. A **Delivery** Process consists of sending one or more products to one or more delivery destinations. Delivery is also used to specify the scheduled transfer of digital assets.

In order to instruct a digital delivery Device to compress or encode the files one can use the input and output **RunList** with **FileSpec**/**@Compression** Attribute, even if no URL is specified. See Section O.7, “DigitalDelivery Examples” on page 1323 for a set of examples.

Resource Properties

Resource referenced by: —

Intent Pairing:

Input of Processes: **Delivery**

Output of Processes: —

Table 8-51: **DeliveryParams** Resource (Sheet 1 of 4)

Name	Data Type	Description
<i>BuyerAccount?</i>	string	Account ID of the buyer with the delivery service.
<i>ContactRefs?</i>	IDREFS	Address and further information of the Contacts responsible for this delivery.
<i>DeliveryCharge?</i>	enumeration	<p>Specifies who pays for a delivery being made by a third party.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Printer</i> – The “<i>Printer</i>” is defined as the person who creates the Resource that is delivered. This includes all suppliers (e.g., binders, prepress suppliers, etc.). <i>Buyer</i> – The customer specified in CustomerInfo. <i>Other</i> – The Contact[@ContactTypes = “<i>DeliveryCharge</i>”].

Table 8-51: DeliveryParams Resource (Sheet 2 of 4)

Name	Data Type	Description
<i>DeliveryID</i> ?	NMTOKEN	<p>Deliveries with the same <i>@DeliveryID</i> are part of the same physical delivery.</p> <p>This attribute allows items from multiple individual XJDF jobs to be delivered in one delivery.</p>
<i>EarliestDuration</i> ?	duration	<p>Specifies the earliest time by which the delivery is to be made relative to the date of the purchase order. Within an RFQ or a Quote, at most one of <i>Earliest</i> or <i>EarliestDuration</i> SHALL be specified. Within a purchase order, <i>EarliestDuration</i> SHALL NOT be specified.</p>
<i>Method</i> ?	NMTOKEN	<p>Specifies a delivery method or Brand (e.g., "ExpressMail" or "InterofficeMail").</p> <p>Values include</p> <ul style="list-style-type: none"> <i>BestWay</i> – The sender decides how to deliver. <i>CompanyTruck</i> <i>Courier</i> <i>CourierNoSignature</i> – a delivery service that does not require receipt stamps at recipient's mailbox and/or mail room. This new value covers the commonly used Japanese 'Mail bin' delivery service. <i>Email</i> <i>ExpressMail</i> <i>InstantMessaging</i> <i>InterofficeMail</i> <i>Local</i> – The files are already in place <i>NetworkCopy</i> – This includes LAN and VPN. <i>Storage</i> – The product is stored by the supplier. <i>OvernightService</i> <i>WebServer</i> – Upload / download from HTTP / FTP server.
<i>Ownership</i> ?	enumeration	<p>Point of transfer of ownership:</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Origin</i> – Ownership of goods is transferred upon leaving point of origin. <i>Destination</i> – Ownership is transferred upon receipt at destination.
<i>RequiredDuration</i> ?	duration	<p>Specifies the time by which the delivery is to be made relative to the date of the purchase order. Within an RFQ or a Quote, at most one of <i>Required</i> or <i>RequiredDuration</i> SHALL be specified. Within a purchase order, <i>RequiredDuration</i> SHALL NOT be specified.</p>
<i>RemoteURLRef</i> ?	IDREF	Reference to a <i>FileSpec</i> that specifies the remote location
<i>ReturnMethod</i> ?	NMTOKEN	<p>Specifies a delivery method for returning the surplus material, and SHALL NOT be specified unless <i>SurplusHandling</i> = "Return".</p> <p>Values include those from: <i>@Method</i></p>
<i>ServiceLevel</i> ?	string	<p>The service level of the specific carrier.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Next Day</i> <i>2nd Day Air</i> <i>Ground</i>.

Table 8-51: DeliveryParams Resource (Sheet 3 of 4)

Name	Data Type	Description
<i>SurplusHandling</i> ?	enumeration	<p>Describes what is to happen with unused or redundant parts of the transfer specified with Transfer = "BuyerToPrinterDeliver" or "BuyerToPrinterPickup" after the Job. The return delivery or pickup address is specified by Contact[contains (@ContactTypes, "SurplusReturn")].</p> <p>Allowed values are</p> <ul style="list-style-type: none"> <i>ReturnWithProduct</i> – The surplus material is delivered back to the customer together with the final product. <i>Return</i> – The surplus material is delivered back independently directly after usage. <i>Pickup</i> – The customer picks up the surplus material. <i>Destroy</i> – The printer destroys the surplus material. <i>PrinterOwns</i> – The surplus material belongs to the printer. <i>Store</i> – The printer has to store the surplus material for future purposes.
<i>TrackingID</i> ?	string	The string that can help in tracking the delivery. The value of the @TrackingID Attribute will depend on the carrier chosen to ship the products.
<i>Transfer</i> ? ²	enumeration	<p>Describes the direction and responsibility of the transfer.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>BuyerToPrinterDeliver</i> – The DeliveryParams describes an input to the Job (e.g., a CD for inserting, a preprinted cover, etc.). In this case, the buyer delivers the merchandise to the printer. The printer is to specify in the quote a special Contact[contains (@ContactTypes, "Delivery")]. The Contact specifies where the buyer is to send the merchandise. <i>BuyerToPrinterPickup</i> – The DeliveryParams describes an input to the Job (e.g., a CD for inserting, a preprinted cover, etc.). In this case, the printer picks up the merchandise. The Contact[contains (@ContactTypes, "Pickup")] specifies where the printer has to pick up the merchandise. <i>PrinterToBuyerDeliver</i> – The DeliveryParams describes an output of the Job. In this case, the printer delivers the merchandise to the buyer. The Contact[contains (@ContactTypes, "Delivery")] specifies where the printer is to send the merchandise. <i>PrinterToBuyerPickup</i> – The DeliveryParams describes an output of the Job. In this case, the buyer picks up the merchandise. The printer is to specify in the quote a special Contact[contains (@ContactTypes, "Pickup")]. The Contact specifies where the buyer is to pick up the merchandise.
<i>DropItem</i> +	element	A delivery MAY consist of multiple products, which are represented by their respective Resource Elements. Each DropItem describes an individual Resource that is part of this delivery.
FileSpec (DeliveryContents) ?	element	Reference to a document to be printed and packaged together with the delivered items.

Table 8-51: DeliveryParams Resource (Sheet 4 of 4)

Name	Data Type	Description
FileSpec (<i>MailingList</i>) ?	element	A FileSpec Resource pointing to a mailing list. The format of the referenced mailing list is implementation dependent.
FileSpec (<i>RemoteURL</i>) ?	element	FileSpec that specifies the remote location of a digital delivery.

8.29.1 Element: DropItem

Table 8-52: DropItem Element (Sheet 1 of 2)

Name	Data Type	Description
<i>ActualAmount</i> ?	integer	Actual amount of items delivered in this drop. Note that this logs the information after the fact in a way that is similar to an Audit . @ <i>ActualAmount</i> was placed here because it is very difficult to map the DeliveryParams structure of individual DropItem Elements to Resource and Audit Elements.
<i>ActualTotalAmount</i> ?	integer	Actual @ <i>TotalAmount</i> of items delivered in this drop. Note that this logs the information after the fact in a way that is similar to an Audit . @ <i>ActualTotalAmount</i> was placed here because it is very difficult to map the DeliveryParams structure of individual DropItem Elements to Resource and Audit Elements.
<i>Amount</i> ?	integer	Specifies the number of Resource ordered. If @ <i>Amount</i> is not specified, defaults to the total amount of the Resource that is referenced by Resource.
<i>OrderedAmount</i> ?	integer	Specifies the original number of Resources ordered. If not specified, defaults to the value of @ <i>Amount</i> . Note that DropItem /@ <i>OrderedAmount</i> corresponds semantically to Resource /@ <i>Amount</i> and DropItem /@ <i>Amount</i> .
<i>Proof</i> ?	NMTOKEN	This DropItem refers to a proof that is specified in a ProofItem of the ContentCheckIntent of this Product Intent Node. Constraint: ContentCheckIntent / ProofItem /@ <i>ProofName</i> SHALL match @ <i>Proof</i> . Exactly one of Resource or @ <i>Proof</i> SHALL be specified.
<i>Resource</i> ?	IDREF	Description of the individual item to be delivered. It can be any kind of Resource or a reference to a ProductList / Product .
<i>TotalAmount</i> ?	integer	Total amount of individual items delivered in this drop. The @ <i>TotalAmount</i> and @ <i>Amount</i> differ if the Resource is a Bundle of multiple Resources. The @ <i>Amount</i> specifies the number of Bundles (e.g., boxes, pallets etc.). Whereas @ <i>TotalAmount</i> specifies the number of final products (e.g., books, magazines etc.).
<i>TotalDimensions</i> ?	Shape	Total dimensions in points of all individual items including packaging delivered in this drop.
<i>TotalVolume</i> ?	double	Total volume in liters of all individual items including packaging delivered in this drop.
<i>TotalWeight</i> ?	double	Total weight in gram of all individual items including packaging delivered in this drop.

Table 8-52: DroplItem Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Unit</i> ?	NMTOKEN	Unit of measurement for the <i>@Amount</i> of the Resource that is referenced by Resource. Default value is from: Resource/ <i>@Unit</i>

8.30 DevelopingParams

DevelopingParams specifies information about the chemical and physical properties of the developing and fixing process for film and plates. Includes details of preheating, postbaking and postexposure.

- **Preheating** is necessary for negative working plates. It hardens the exposed areas of the plate to make it durable for the following developing process. The stability and uniformity of the preheat temperature influence the evenness of tints and the run length of the plate on press.
- **Postbaking** is an optional process of heating that is applied to most polymer plates to enhance the run length of the plate. A factor 5 to 10 can be gained compared to plates that are not postbaked.
- **Postexposure** is an optional exposure process for photopolymer plates to enhance the run length of the plate. A factor of 5 to 10 can be gained compared with plates that are not postexposed.

Note: Postbaking and postexposure are mutually exclusive.

Resource Properties

Resource referenced by: —

Input of Processes: *ImageSetting*

Output of Processes: —

Table 8-53: DevelopingParams Resource

Name	Data Type	Description
<i>PreHeatTemp</i> ?	double	Temperature of the preheating Process in °C.
<i>PreHeatTime</i> ?	duration	Duration of the preheating Process.
<i>PostBakeTemp</i> ?	double	Temperature of the postbaking Process in °C.
<i>PostBakeTime</i> ?	duration	Duration of the postbaking Process. <i>@PostBakeTime</i> SHALL NOT be specified if <i>@PostExposeTime</i> is present.
<i>PostExposeTime</i> ?	duration	Duration of the postexposing Process. <i>@PostExposeTime</i> SHALL NOT be specified if <i>@PostBakeTime</i> is present.

8.31 Device

Information about a specific Device. This can include information about the Devices capabilities. For more information, see Section 8.1, “Resource”.

Resource Properties

Resource referenced by: AuditStatus_DeviceFilter, DeviceInfo, Queue, QueueFilter, **DieLayout**, **DieLayoutProductionParams**/ConvertingConfig, **InkZoneCalculationParams**, **PrintCondition**, **StrippingParams**

Input of Processes: *Any Process*

Output of Processes: —

Table 8-54: Device Resource (Sheet 1 of 3)

Name	Data Type	Description
<i>CostCenterID</i> ?	string	MIS cost center ID.

Table 8-54: Device Resource (Sheet 2 of 3)

Name	Data Type	Description
<i>DeviceClass</i> ?	NMOKENS	<p>Indicates type of device. Multiple NMOKENS are used to describe integrated devices with multiple classes.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>CaseMaker</i> <i>Cutter</i> <i>DieCutter</i> <i>EndsheetFeeder</i> <i>Folder</i> <i>Gatherer</i> <i>GathererBinder</i> <i>Hardcover</i> <i>HardcoverBookLine</i> <i>Inserter</i> <i>Jacketer</i> <i>PerfectBinder</i> <i>PerfectBinderLine</i> <i>PlateSetter</i> <i>PrintDelivery</i> <i>PrintingPress</i> <i>Stacker</i> <i>Stitcher</i> <i>ThreadSewer</i> <i>Trimmer</i> <i>WideFormatPrinter</i>
<i>DeviceID</i> ?	NMOKEN	Identifier of the Device. The <i>@DeviceID</i> SHALL be unique within the workflow. <i>@DeviceID</i> SHALL be the same over time for a specific Device instance (i.e., SHALL survive reboots). If the Device sends XJMF Messages, this value SHALL also be used for JMF XJMF@DeviceID and @DeviceID of the specific messages.
<i>DeviceType</i> ?	string	Manufacturer type ID, including a revision stamp. Type of the Device. Used for grouping and filtering of Devices
<i>Directory</i> ?	URL	Defines a directory where the URLs that are associated with this Device can be located. If <i>@Directory</i> is specified, it SHALL be an Absolute URI [RFC3986] that implicitly also specifies a Base URI which is used to resolve any relative URL of Device . See Appendix K, “Resolving RunList/@Directory and FileSpec/@URL URI References” on page 1255 and [FileURL].
<i>ICSVersions</i> ?	NMOKENS	<p>CIP4 Interoperability Conformance Specification (ICS) Versions that this Device complies with.</p> <p>Values include those from: JDF/@<i>ICSVersions</i> (Table 3-4, “JDF XJDF Node” on page 55).</p>

Table 8-54: Device Resource (Sheet 3 of 3)

Name	Data Type	Description
<i>JDFVersions</i> ?	JDFJMVFersions	Whitespace separated list of supported XJDF versions that this Device supports (e.g., "1.3 2.0" specifies that both the 1.3 and 2.0 versions are supported).
<i>JMFURL</i> ?	URL	URL of the Device port that will accept XJMF Messages. A Controller that manages a Device MAY specify its own @ <i>JMFURL</i> when responding to KnownDevices Messages. This is how a Controller inserts itself as the manager for a Device.
<i>KnownLocalizations</i> ?	languages	A list of all language codes supported by the Device for localization. If not specified, then the Device supports no localizations.
<i>Manufacturer</i> ?	string	Manufacturer name.
<i>ManufacturerURL</i> ?	URL	Web site for manufacturer.
<i>ModelDescription</i> ?	string	Long description for end user.
<i>ModelName</i> ?	string	Model name.
<i>ModelNumber</i> ?	string	Model number.
<i>ModelURL</i> ?	URL	Web site for model.
<i>Packaging</i> ?	enumerations	<p>List of packaging methods supported for job submission with SubmitQueueEntry, ResubmitQueueEntry and ReturnQueueEntry.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> ZIP – ZIP packaging of XJMF, XJDF and digital assets is supported. See Section 11.6, JDF XJDF Packaging for details. XML – unpackaged XML is supported.
<i>PresentationURL</i> ?	URL	@ <i>PresentationURL</i> specifies a URL to a Device-provided user interface for configuration, status, etc. For instance, if the Device has an embedded Web server, this is a URL to the configuration page hosted on that Web server.
<i>SecureJMFURL</i> ?	URL	URL of the Device port that will accept XJMF Messages via the "https" protocol.
<i>SerialNumber</i> ?	string	Serial number of the Device.
<i>UPC</i> ?	URL	Universal Product Code for the Device. A 12-digit, all-numeric code that identifies the consumer package. Managed by the Uniform Code.
<i>URLSchemes</i> ?	NMTOKENS	<p>List of schemes supported in for retrieving XJDF files. If not specified, the Controller does not support retrieving XJDF files from remote URLs.</p> <p>Values include:</p> <ul style="list-style-type: none"> file – The file scheme according to [RFC1738] and [RFC3986]. ftp – FTP (File Transfer Protocol) http – HTTP (Hypertext Transport Protocol) https – HTTPS (Hypertext Transport Protocol — Secure)
<i>IconList</i> ?	element	List of locations of icons that can be used to represent the Device.
<i>Location</i> ?	element	Description of the Device location.
<i>Module</i> *	element	Individual Modules that are represented by this Device .

8.31.1 Element: IconList

The IconList is a list of individual icon descriptions.

Table 8-55: IconList Element

Name	Data Type	Description
Icon +	element	Individual icon description.

8.31.2 Element: Icon

An Icon represents a Device in the user interface.

Table 8-56: Icon Element

Name	Data Type	Description
<i>BitDepth</i> ?	integer	Bit depth of one color.
<i>IconUsage</i> ?	enumerations	<p>Definition of the <i>DeviceInfo/@DeviceStatus</i> of the Device that this Icon represents.</p> <p>Any combination of values are allowed:</p> <p>Default value is: a list of all values (i.e., no limit on Icon use).</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Unknown</i> – No link to the Device exists <i>Idle</i> <i>Down</i> <i>Setup</i> <i>Running</i> <i>Cleanup</i> <i>Stopped</i> <p>Note: The meaning of the individual enumerations is described in the <i>DeviceInfo</i> Message Element. See Section 5.8.3, “KnownDevices”.</p>
<i>URLSchemes</i> ?	NMTOKENS	<p>List of schemes supported in for retrieving XJDF files. If not specified, the Controller does not support retrieving XJDF files from remote URLs.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>file</i> – The file scheme according to [RFC1738] and [RFC3986] <i>ftp</i> – FTP (File Transfer Protocol) <i>http</i> – HTTP (Hypertext Transport Protocol) <i>https</i> – HTTPS (Hypertext Transport Protocol — Secure)
<i>Size</i> ?	XYPair	Height and width of the icon.
<i>FileSpec</i> ?	element	Details of the file containing the icon data.

8.31.3 Element: Module

A Module represents a physical Machine or part of a Device.

Table 8-57: Module Element (Sheet 1 of 2)

Name	Data Type	Description
<i>DeviceType</i> ?	string	Manufacturer type ID, including a revision stamp.

Table 8-57: Module Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Manufacturer</i> ?	string	Manufacturer name.
<i>ManufacturerURL</i> ?	URL	Web site for manufacturer.
<i>ModelDescription</i> ?	string	Long description for end user.
<i>ModelName</i> ?	string	Model name.
<i>ModelNumber</i> ?	string	Model number.
<i>ModelURL</i> ?	URL	Web site for model.
<i>ModuleID</i> ?	NMTOKEN	Name of the Module. This is a unique identifier within the workflow. <i>@ModuleID</i> SHALL be the same over time for a specific Device instance (i.e., SHALL survive reboots). If multiple logical Devices share a physical Module, <i>@ModuleID</i> SHALL be identical. <i>@ModuleID</i> SHOULD be used to specify Machines that comprise a Device .
<i>ModuleType</i> ?	NMOKEN	Type of Module. Values include those from: Section C.2, “ModuleType Supported Strings” on page 1170. Note: the allowed values depend on the type of Device. Each type of Device has a separate table of values.
<i>SerialNumber</i> ?	string	Serial number of the Device.

8.32 DieLayout

DieLayout represents a die layout described in an external file. This Resource is also used as the input for the actual die making process and is also used in **Stripping**. The external file is by preference a DDES3 file (ANSI® IT8.6-2002). The usage of other files like CFF2, DDES2, DXF or proprietary formats is not excluded but MAY have a negative impact on interoperability.

Resource Properties

Resource referenced by: **BinderySignature, ShapeCuttingParams**

Input of Processes: **DieDesign, DieMaking**

Output of Processes: **DieDesign, DieLayoutProduction**

Table 8-58: DieLayout Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>CutBox</i> ?	rectangle	A rectangle describing the bounding box of all cut lines in the DieLayout . This is sometimes referred to as the knife to knife dimensions of the DieLayout . If the position on the Media is not known, the lower left SHOULD be set to 0 0.
<i>DieSide</i> ?	enumeration	Determines the die side for which the DieLayout is made. Allowed values are: <i>Up</i> – the DieLayout is made with the knives pointing upwards. <i>Down</i> – the DieLayout is made with the knives pointing downwards.
<i>MediaSide</i> ?	enumeration	Determines the printing side for which the DieLayout is made. Allowed values are: <i>Front</i> – for a box this corresponds to the outside of a box. <i>Back</i> – for a box this corresponds to the inside of a box.

Table 8-58: DieLayout Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>Rotated</i> ?	boolean	Indicates if some of the structural designs are oriented cross grain/flute in the layout.
<i>Waste</i> ?	double	The percent of the material that is wasted. Inner waste (i.e., cut out windows are not included in the waste).
Device *	element	The Devices for which this DieLayout was made (printing press and die cutter). Typically only the type of Device would be used (e.g., the model of the die cutter).
FileSpec ?	element	Reference to an external URL that represents the die. This is typically a CAD design file.
Media ?	element	Media for which this DieLayout was intended. The Media description defines important design parameters as the type of Media , dimensions, grain direction or flute direction.
RuleLength *	element	Elements describing the length of die rules for the different types of rules. Each RuleLength Element describes the accumulated length of all rules of a certain type.
Station *	element	Description of the stations in a DieLayout . One Station produces one shape type.

8.32.1 Element: RuleLength

Elements describing the length of die rules for the different types of rules. Each **RuleLength** Element describes the accumulated length of all rules of a certain type.

Table 8-59: RuleLength Element

Name	Data Type	Description
<i>DDESCutType</i>	integer	Type of rule. Values include: a number between "0" and "999" corresponding to a line type as defined in DDES.
<i>Length</i>	double	Accumulated length of the rules of this type in the DieLayout (pt).

8.32.2 Element: Station

Description of the stations in a **DieLayout**. One station produces one shape type. On **Station** element MAY represent multiple identical one-up stations on an N-up **DieLayout**.

Table 8-60: Station Element (Sheet 1 of 2)

Name	Data Type	Description
<i>BinderySignatureIDs</i> ?	NMTOKENS	The list of <i>@AssemblyIDs</i> of the graphic elements that are processed by this Station . If specified, <i>@BinderySignatureIDs</i> SHALL list the BinderySignature / <i>@BinderySignatureID</i> of all BinderySignatures that are processed by this Station .
<i>StationAmount</i> ?	integer	The number of stations in the DieLayout with this <i>@StationName</i> .
<i>StationName</i> ?	string	The name of the 1-up design in the DieLayout .

Table 8-60: Station Element (Sheet 2 of 2)

Name	Data Type	Description
<code>ShapeDef ?</code>	element	The <code>ShapeDef</code> corresponding to this station in the <code>DieLayout</code> .

8.33 DieLayoutProductionParams

Parameters for the die layout.

Resource Properties

Resource referenced by: —

Input of Processes: *DieLayoutProduction*

Output of Processes: —

Table 8-61: DieLayoutProductionParams Resource

Name	Data Type	Description
<code>Estimate ?</code>	boolean	Determines if the Process runs in estimate mode or not. When in estimate mode multiple solutions are generated.
<code>Position ?</code>	Anchor	The position of layout on the sheet.
<code>ConvertingConfig *</code>	element	A <code>ConvertingConfig</code> Element describes a range of Sheet sizes that can be taken into account to create a new <code>DieLayout</code> . Typically a <code>ConvertingConfig</code> will correspond to 1 combination of printing press and further finishing equipment such as die cutters.
<code>RepeatDesc ?</code>	element	Step and repeat parameters for a set of <code>ShapeDef</code> .

8.33.1 Element: RepeatDesc

The `RepeatDesc` Element describes the layout specs for a `ShapeDef`.

Table 8-62: RepeatDesc Element (Sheet 1 of 2)

Name	Data Type	Description
<code>AllowedRotate ?</code>	enumeration	Allowed methods to rotate structural designs in with respect to grain/flute. Allowed values are: <i>None</i> – No Rotation at all. <i>Grain</i> – 0° or 180° Rotation. <i>MinorGrain</i> – device dependent small rotations that retain the general grain direction (e.g., +/- 10°). <i>CrossGrain</i> – Cross grain rotations (e.g., 90° are acceptable).
<code>GutterX ?</code>	double	Gutter between columns (see also <code>@GutterX2</code>)
<code>GutterX2 ?</code>	double	Secondary gutter between columns. When the <code>@LayoutStyle = "Reverse2ndColumn"</code> , the gutter between columns (2n+1) and (2n+2) is <code>@GutterX</code> and between columns (2n+2) and (2n+3) is <code>@GutterX2</code> . When <code>@GutterX2</code> is not specified <code>@GutterX2 = @GutterX</code> . See Figure 8-31, “RepeatDesc/ <code>@GutterX2</code> and <code>@GutterY2</code> : Secondary Gutters,” on page 643.
<code>GutterY ?</code>	double	Gutter between rows (see also <code>@GutterY2</code>).

Table 8-62: RepeatDesc Element (Sheet 2 of 2)

Name	Data Type	Description
<i>GutterY2</i> ?	double	Secondary gutter between rows. When the <i>@LayoutStyle</i> = "Reverse2ndRow" the gutter between rows (2n+1) and (2n+2) is <i>@GutterY</i> and between rows (2n+2) and (2n+3) <i>@GutterY2</i> . When <i>@GutterY2</i> is not specified <i>@GutterY2</i> = <i>@GutterY</i> . See Figure 8-31, "RepeatDesc/@GutterX2 and @GutterY2: Secondary Gutters," on page 643.
<i>LayoutStyle</i> ?	NMTOKENS	The allowed styles for the Layout Values include: <i>StraightNest</i> <i>Reverse2ndRow</i> <i>Reverse2ndRowAligned</i> <i>Reverse2ndColumn</i> <i>Reverse2ndColumnAligned</i> Note: for diagrams of the above values, see Figure 8-25, "Basic Shape for RepeatDesc/@LayoutStyle Examples," on page 640 and the following five figures
<i>OrderQuantity</i> ?	integer	The order quantity for the 1-up for which this layout will be optimized. This information needs to be present when a Layout is being made for more than 1 ShapeDef .
<i>UseBleed</i> ?	boolean	If true, the print bleed defined in the structural design is used to calculate the layout. If false, the outer cut is used.
ShapeDef *	element	ShapeDef Resources describing the different 1-up structural designs to be stepped and repeated on the Media .

The following Figure shows the basic shape for subsequent Figures. that relate to RepeatDesc.

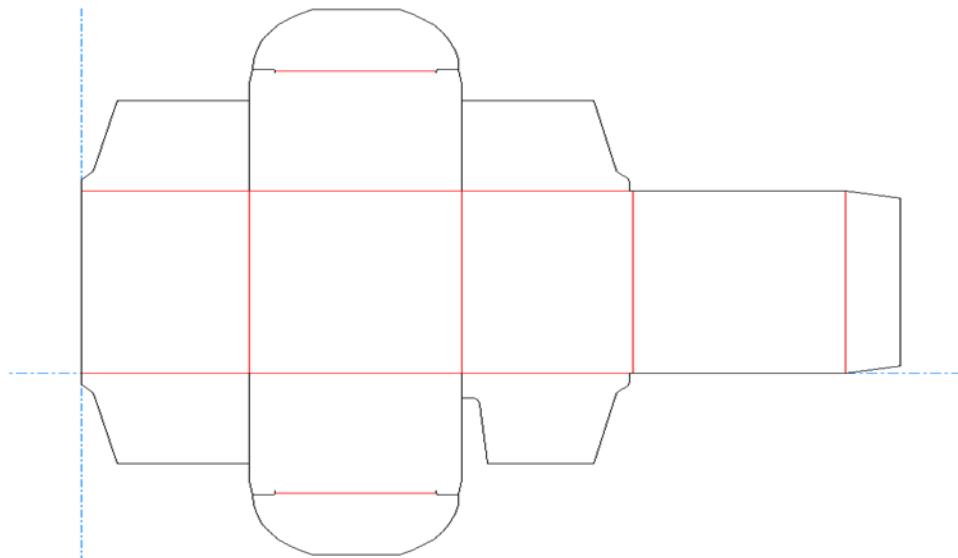
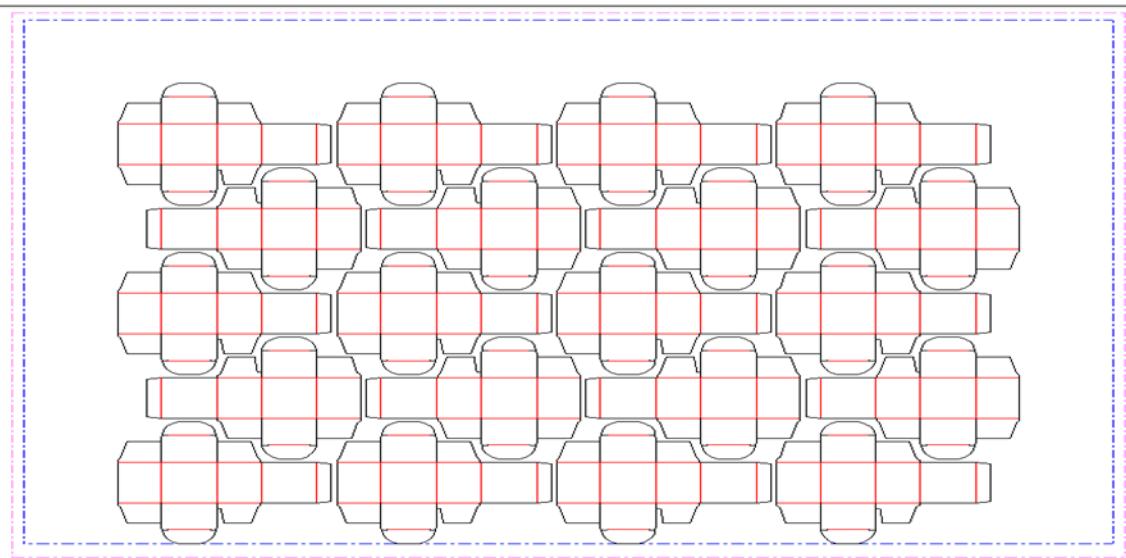
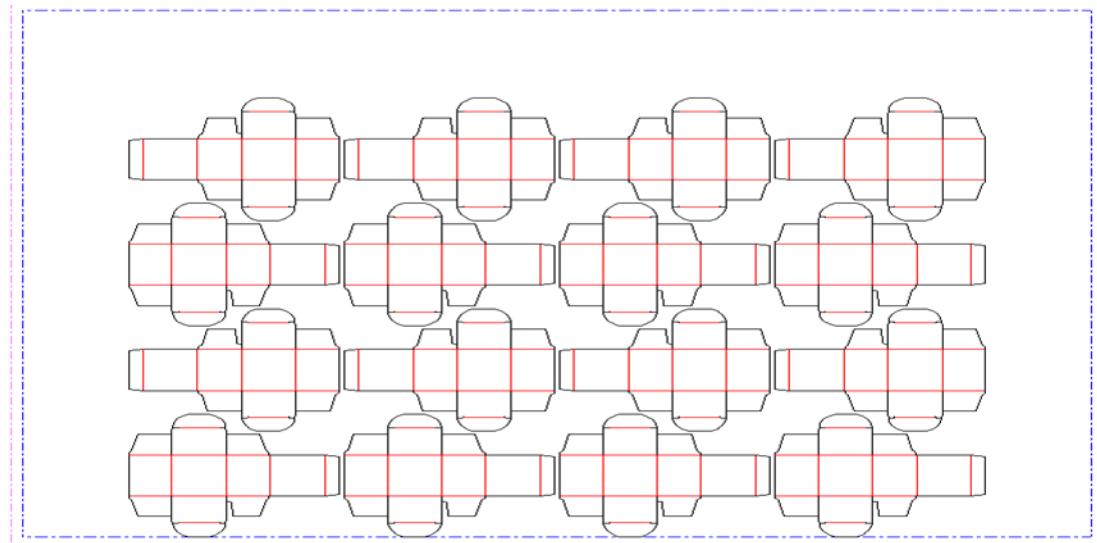
Figure 8-19: Basic Shape for RepeatDesc/@LayoutStyle Examples

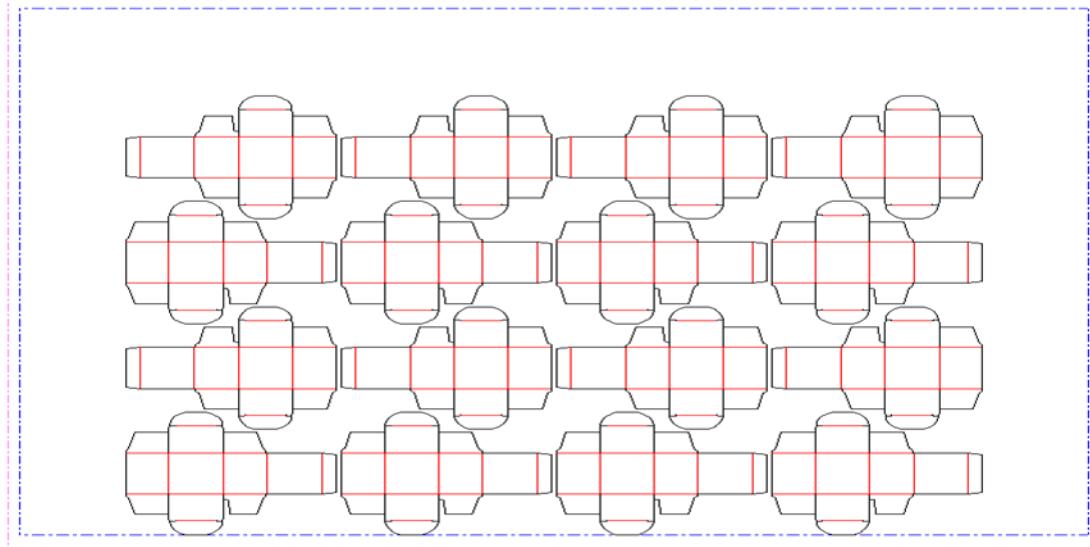
Figure 8-20: RepeatDesc/@LayoutStyle = "StraightNest"

In the following Figure, 1-ups on even rows are rotated 180 degrees. Even rows are shifted horizontally and vertically to obtain optimal nesting.

Figure 8-21: RepeatDesc/@LayoutStyle = "Reverse2ndRow"

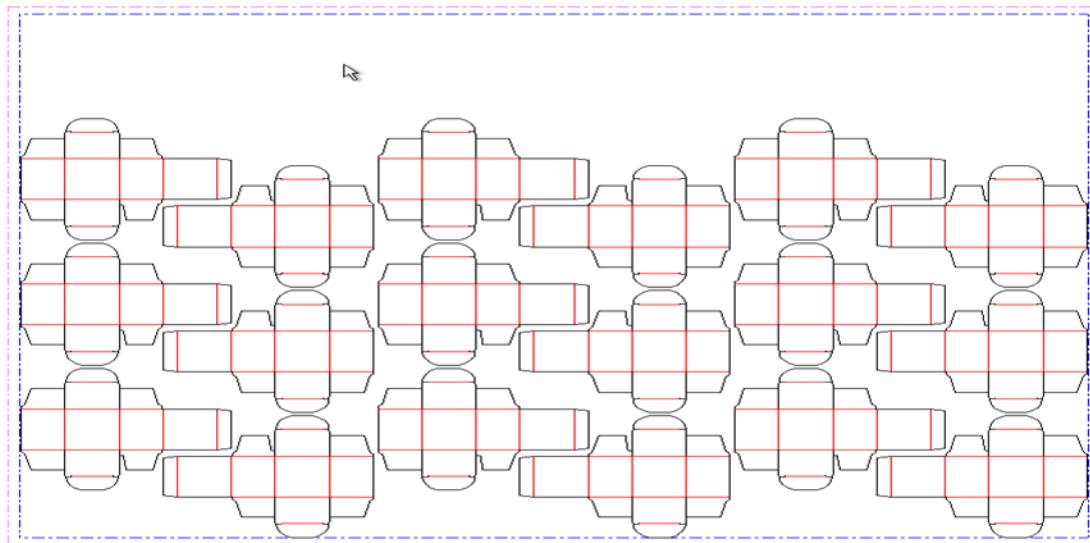
In the following Figure, 1-ups on even rows are rotated 180 degrees. Even rows are shifted vertically to obtain optimal nesting. The even rows are not shifted horizontally. (Left and right edges are aligned between rows)

Figure 8-22: RepeatDesc/@LayoutStyle = "Reverse2ndRowAligned"

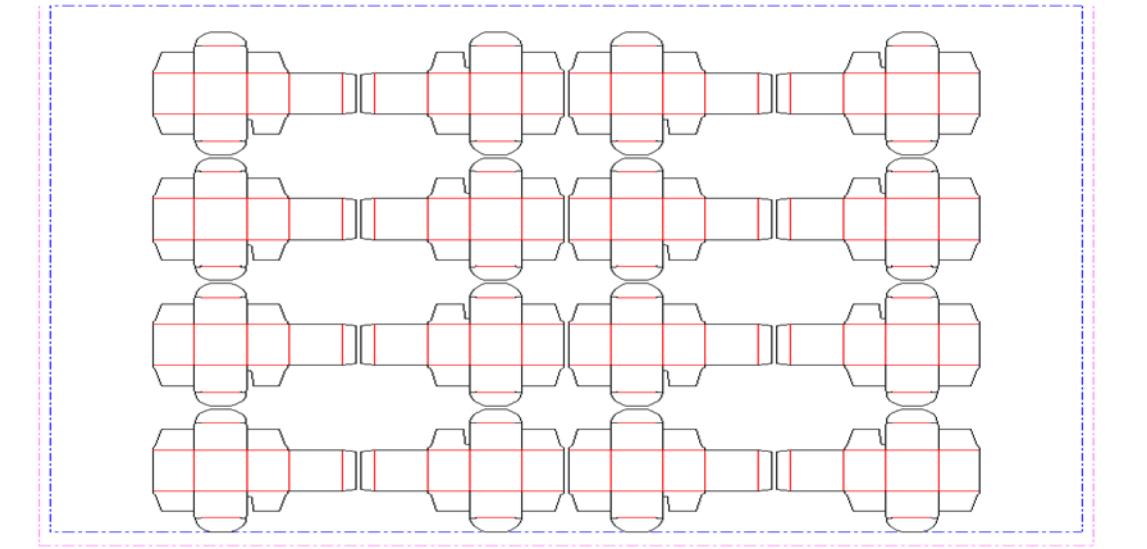


In the following Figure, 1-ups on even columns are rotated 180 degrees. Even columns are shifted vertically and horizontally to obtain optimal nesting.

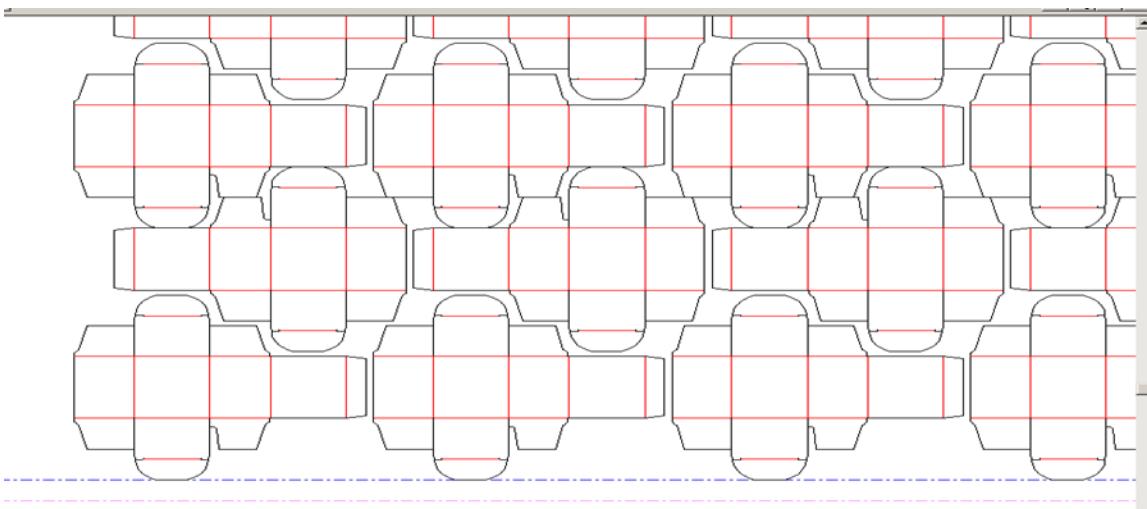
Figure 8-23: RepeatDesc/@LayoutStyle = "Reverse2ndColumn"



In the following Figure, 1-ups on even columns are rotated 180 degrees. Even columns are shifted horizontally to obtain optimal nesting. No vertical shifting of even columns is done (top and bottom edges are aligned between columns).

Figure 8-24: RepeatDesc/@LayoutStyle = "Reverse2ndColumnAligned"

In the following Figure, @LayoutStyle = "Reverse2ndRow", @GutterY = "15", @GutterY2 = "0".

Figure 8-25: RepeatDesc/@GutterX2 and @GutterY2: Secondary Gutters

8.34 DigitalMedia

This Resource represents a processed removable digital media-based Resource such as tape or removable disk.

Resource Properties

Resource referenced by: DropItem

Input of Processes: —

Output of Processes: —

Table 8-63: DigitalMedia Resource

Name	Data Type	Description
<i>Capacity</i> ?	integer	Size of the digital media in megabytes.
<i>MediaLabel</i> ?	string	Electronic label of the media.
<i>MediaType</i>	NMOKEN	<p>The digital media type. Values include:</p> <p><i>CD</i> – Recordable compact disc.</p> <p><i>DAT</i> – DAT tape backup media.</p> <p><i>DLT</i> – DLT tape backup media.</p> <p><i>DVD</i> – DVD disc.</p> <p><i>Exabyte</i> – Exabyte tape backup media.</p> <p><i>HardDrive</i> – Removable hard drives from a rack.</p> <p><i>Jaz</i> – Jaz removable disk drive.</p> <p><i>Optical</i> – Optical removable disk drive. Excluding CDs and DVDs.</p> <p><i>SDCard</i></p> <p><i>SSD</i></p> <p><i>Tape</i> – Tape backup media. Use only when the explicit tape type is not listed here.</p> <p><i>USBDrive</i></p> <p><i>Zip</i> – Zip removable disk drive.</p>
<i>MediaTypeDetails</i> ?	string	The digital media type details — could be vendor or model name. For example: " <i>8mm</i> " or " <i>VHS</i> " for tape media.
<i>RunList</i> ?	element	Link to the relevant files on the media. The URLs specified in RunList/ FileSpec/@URL SHOULD be relative paths to the media's mount point.

8.35 DigitalPrintingParams

This Resource contains Attributes and Elements used in executing the **DigitalPrinting** Process. The **@PrintingType** Attribute in this Resource defines two types of printing: "*SheetFed*" and "*WebFed*". The principal difference between them is the shape of the paper each is equipped to accept. Presses that execute "*WebFed*" Processes use substrates that are continuous and cut after printing is accomplished. Most newspapers are printed on Web Presses. "*SheetFed*" printing, on the other hand, accepts precut substrates.

8.35.1 Coordinate systems in DigitalPrinting

Figure 2-13 in Section 2.5, “Coordinate Systems in JDF XJDF” defines the coordinate system for **ConventionalPrinting** and **DigitalPrinting**. Note that the paper feed direction of the idealized Process is towards the X-axis which corresponds to bottom edge first.

Properties

referenced by: —

Input of Processes: **DigitalPrinting**

Output of Processes: —

Table 8-64: DigitalPrintingParams Resource (Sheet 1 of 3)

Name	Data Type	Description
<i>Collate</i> ?	enumeration	<p>Determines the sequencing of the Sheets in the document and the documents in the Job when multiple copies of a document or a Job are requested as output. Document copies can be requested by specifying RunList/@DocCopies and Job copies can be requested by specifying the output Component/@Amount.</p> <p>Allowed values are:</p> <p><i>None</i> – Do not collate Sheets in the document or document(s) in the Job.</p> <p><i>Sheet</i> – Collate the Sheets in each document; do not collate the documents in the Job. The result of "<i>Sheet</i>" and "<i>SheetAndSet</i>" is the same when there is one document in the set. The result of "<i>Sheet</i>" and "<i>SheetSetAndJob</i>" is the same when there is one document in the set and one set in the Job.</p> <p><i>SheetAndSet</i> – Collate the Sheets in the document and collate the documents in the set. Do not collate the sets in the Job. The result of "<i>SheetAndSet</i>" and "<i>SheetSetAndJob</i>" is the same when there is one set in the Job.</p> <p><i>SheetSetAndJob</i> – Collate the Sheets in the document and collate the documents in the set and collate the sets in the Job.</p> <p>Example: two documents, A and B, each have two Sheets, A1, A2 and B1, B2. The number of document copies requested is one for both documents and the number of Job copies requested is three (Component/@Amount = 3). The Job contains no Document Set boundaries.</p> <p>If @Collate = "None", the Sheet order will be: A1A1A1 A2A2A2 B1B1B1 B2B2B2</p> <p>If @Collate = "Sheet", the Sheet order will be: A1A2 A1A2 A1A2 B1B2 B1B2 B1B2</p> <p>If @Collate = "SheetAndSet" or "SheetSetAndJob", the Sheet order will be: A1A2 B1B2 A1A2 B1B2 A1A2 B1B2</p>
<i>ManualFeed</i> ?	boolean	Indicates whether the media will be fed manually.
<i>OutputBin</i> ?	NMTOKENS	<p>Specifies the bin or bins to which the finished documents SHALL be output. If multiple values are provided, the output bins SHALL be filled in sequence. See @StackAmount.</p> <p>Values include those from: Table C-21, “Input Tray and Output Bin Names” on page 1179.</p>

Table 8-64: DigitalPrintingParams Resource (Sheet 2 of 3)

Name	Data Type	Description
<i>PageDelivery</i> ?	enumeration	<p>Indicates how pages are to be delivered to the output bin or finisher.</p> <p>Note: these values refer to the orientation of the entire stack being output from the press, not individual sheets. For example, "<i>SameOrderFaceDown</i>" means that the stack can be picked up and turned over to find the output sheets in the same order as the input RunList with the first page on top facing up</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>FanFold</i> – The output is alternating face-up, face down. <i>SameOrderFaceUp</i> – Order as defined by the RunList, with the "<i>Front</i>" sides of the media up and the first Sheet on top. <i>SameOrderFaceDown</i> – Order as defined by the RunList, with the "<i>Front</i>" sides of the media down and the first Sheet on the bottom. <i>ReverseOrderFaceUp</i> – Sheet order reversed compared to "<i>SameOrderFaceUp</i>", with the <i>Front</i> sides of the media up and the last Sheet on top. <i>ReverseOrderFaceDown</i> – Sheet order reversed compared to "<i>SameOrderFaceDown</i>", with the <i>Front</i> sides of the media down and the last Sheet on the bottom.
<i>SheetLay</i> ?	enumeration	<p>Lay of input media. Reference edge of where paper is placed in feeder.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Left</i> <i>Right</i> <i>Center</i>

Table 8-64: DigitalPrintingParams Resource (Sheet 3 of 3)

Name	Data Type	Description
<i>Sides</i> ?	enumeration	<p>Indicates whether the <i>ByteMap</i> SHALL be imaged on one or both sides of the media. If the <i>RunList(Surface)</i> input to DigitalPrinting is Partitioned by <i>@Side</i> (either explicitly or implicitly using the <i>RunList/@SheetSides</i> Attribute), then the input <i>RunList</i> provides a binding of front and back surfaces to sheets. If <i>@Sides</i> = "<i>OneSidedFront</i>" or "<i>OneSidedBack</i>", then that binding is ignored and one surface is imaged per sheet. If the <i>RunList (Surface)</i> does not provide the binding of surfaces to sides, then the <i>@Sides</i> Attribute specifies the binding to be applied. When a different value for this Attribute is encountered, it SHALL force a new Sheet. However, when the same value for this Attribute is restated for consecutive pages, it is the same as if that restatement was not present.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>OneSidedBack</i> <i>OneSidedFront</i> <i>TwoSided</i> <p>Note: The orientation of the front pages relative to back pages SHOULD be completely defined in the explicit or implied imposition Layout.</p>
<i>StackAmount</i> ?	integer	Specifies the maximum sheet count before switching to the next stacker in the list of <i>@OutputBin</i> values.
ApprovalParams ?	element	Details of the direct approval Process, when <i>@DirectProofAmount</i> > 0.

8.36 ContentEmbossingParams

This Resource contains Attributes and Elements used in executing the **Embossing** Process. The **Embossing** can also be used to model a foil stamping Process.

Resource Properties

Resource referenced by: —

Intent Pairing: **EmbossingIntent**

Input of Processes: **Embossing**

Output of Processes: —

Table 8-65: EmbossingParams Resource

Name	Data Type	Description
<i>ModuleID</i> ?	NMTOKEN	Identifier of the embossing Module in the Press. See ConventionalPrintingParams .
Emboss *	element	One Emboss Element is specified for each impression.

8.36.1 Element: Emboss

Table 8-66: Emboss Element (Sheet 1 of 2)

Name	Data Type	Description
<i>BrailleText</i> ?	string	Text to be embossed as Braille text if <i>@EmbossingType</i> = "Braille"
<i>Direction</i>	enumeration	<p>The direction of the image.</p> <p>Allowed values are:</p> <p><i>Both</i> – Both debossing and embossing in one stamp.</p> <p><i>Flat</i> – The embossing foil is applied flat. Used for foil stamping.</p> <p><i>Raised</i> – Embossing.</p> <p><i>Depressed</i> – Debossing.</p>
<i>EdgeAngle</i> ?	double	The angle of a beveled edge in degrees. Typical values are an angle of: 30, 40, 45, 50 or 60 degrees. If <i>@EdgeAngle</i> is specified, <i>@EdgeShape</i> = "Beveled" SHALL be specified.
<i>EdgeShape</i> ?	enumeration	<p>The transition between the embossed surface and the surrounding media can be rounded or beveled (angled).</p> <p>Allowed values are:</p> <p><i>Rounded</i></p> <p><i>Beveled</i></p>
<i>EmbossingType</i>	enumeration	<p>Allowed values are:</p> <p><i>BlindEmbossing</i> – Embossed forms that are not inked or foiled. The color of the image is the same as the paper.</p> <p><i>Braille</i> – 6 dot Braille embossing.</p> <p><i>EmbossedFinish</i> – The overall design or pattern impressed in laminated paper when passed between metal rolls engraved with the desired pattern. Produced on a special embossing to create finishes such as linen.</p> <p><i>FoilEmbossing</i> – Combines embossing with foil stamping in one single impression.</p> <p><i>FoilStamping</i> – Using a heated die to place a metallic or pigmented image from a coated foil on the paper.</p> <p><i>RegisteredEmbossing</i> – Creates an embossed image that exactly registers to a printed image.</p>
<i>Height</i> ?	double	The height of the levels. This value specifies the <i>vertical</i> distance between the highest and lowest point of the stamp, regardless of the value of <i>@Direction</i> .
<i>ImageSize</i> ?	XYPair	The size of the bounding box of one single image.
<i>Level</i> ?	enumeration	<p>The level of embossing.</p> <p>Allowed values are:</p> <p><i>SingleLevel</i></p> <p><i>MultiLevel</i></p> <p><i>Sculpted</i></p>
<i>Position</i> ?	XYPair	Position of the lower left corner of the bounding box of the embossed image in the coordinate system of the Component .

Table 8-66: Emboss Element (Sheet 2 of 2)

Name	Data Type	Description
IdentificationField ?	element	If <i>@EmbossingType</i> = "Braille", IdentificationField describes the content of the Braille Element.
Tool ?	element	The tool used to make the embossing described by this Element.

8.37 Resource **EndSheetGluingParams**

This Resource describes the Attributes and Elements used in executing the **EndSheetGluing** Process.

Resource Properties

Resource referenced by: —

Intent Pairing: **BindingIntent**

Input of Processes: **EndSheetGluing**

Output of Processes: —

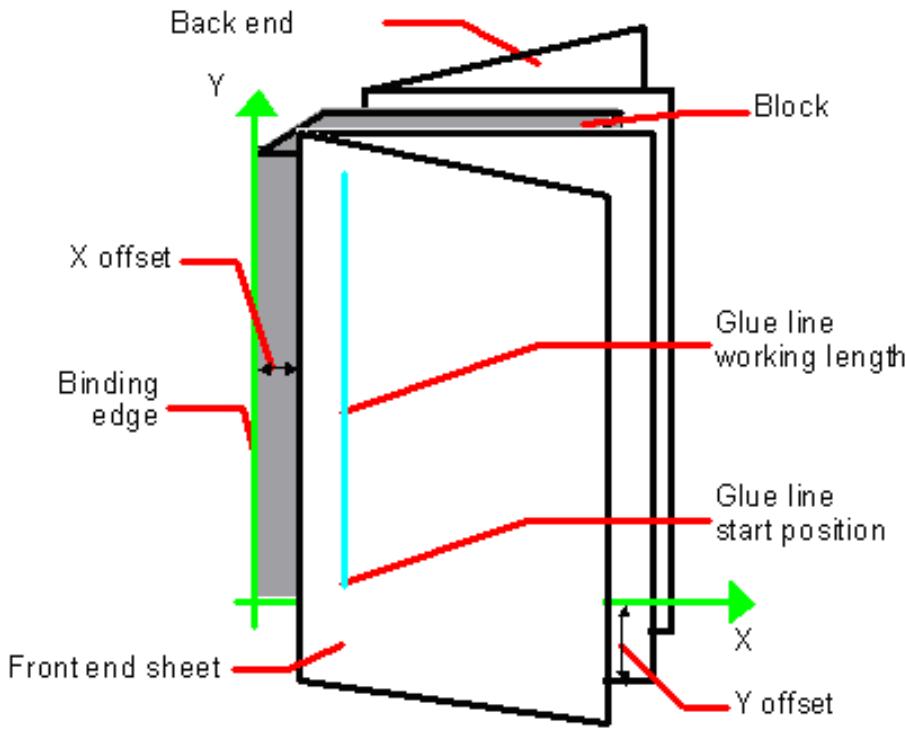
Table 8-67: EndSheetGluingParams Resource

Name	Data Type	Description
EndSheet(Front) ?	element	Information about the front-end Sheet. The <i>@Side</i> Attribute of this Element SHALL be "Front".
EndSheet(Back) ?	element	Information about the back-end Sheet. The <i>@Side</i> Attribute of this Element SHALL be "Back".

8.37.1 Element: **EndSheet**

Table 8-68: EndSheet Element

Name	Data Type	Description
Side	enumeration	Location of the end Sheet. Allowed values are: <i>Front</i> <i>Back</i>
GlueLine ?	element	Description of the glue line.

Figure 8-26: Parameters and coordinate system used for end-Sheet gluing

The process coordinate system is defined as follows: The Y-axis is aligned with the binding edge of the book block. It increases from the registered edge to the edge opposite to the registered edge. The X-axis is aligned with the registered edge. It increases from the binding edge to the edge opposite the binding edge (i.e., the product front edge).

8.38 ExposedMedia

This Resource represents a processed **Media**-based Resource such as film or plate. The *@ExternalID* attribute in the parent **Resource** element SHALL be unique within the workflow.

Resource Properties

Resource referenced by: —

Input of Processes: *Bending, ConventionalPrinting, DigitalPrinting, ImageSetting, PreviewGeneration, Varnishing*

Output of Processes: *Bending, ImageSetting*

Table 8-69: ExposedMedia Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>ContentDataRefs</i> ?	IDREFS	Specification of content metadata for pages described by this ExposedMedia .
<i>MediaRef</i>	IDREF	Describes media specifics such as size and type.

Table 8-69: ExposedMedia Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>PlateType</i> ?	enumeration	Specifies whether a plate is exposed or a dummy plate. Allowed values are: <i>Exposed</i> – The plate has been imaged. <i>Dummy</i> – Specifies a dummy plate that has not been imaged. Usually, dummy plates are only needed on newspaper-Web Presses.
<i>Polarity</i> ?	boolean	" <i>False</i> " if the media contains a negative image.
<i>PunchType</i> ?	NMTOKEN	Name of the registration punch scheme. If not specified, no holes are punched. Values include: <i>Bacher</i> <i>Stoesser</i>
<i>Resolution</i> ?	XYPair	Resolution of the output.

8.39 FeedingParams

The parameters for any XJDF Feeder processing Device.

Resource Properties

Resource referenced by: —

Input of Processes: **Feeding**

Output of Processes: —

Table 8-70: FeedingParams Resource

Name	Data Type	Description
<i>Feeder</i> *	element	Defines the specifics of an individual Feeder . If a Component or Media from the Input Resource list is not referenced from a Feeder in this list, a system defined Feeder will be used.
<i>CollatingItem</i> *	element	Defines the collating sequence of the input Component (s). If a CollatingItem is not defined, then one Component in the order of input Resource list is consumed.

8.39.1 Element: Feeder

Table 8-71: Feeder Element (Sheet 1 of 2)

Name	Data Type	Description
<i>AlternatePositions</i> ?	IntegerList	Positions of alternate feeders including the feeder specified in <i>@Position</i> on a feeding chain. Alternate feeders share the load according to the policy defined in <i>@FeederSynchronization</i> . If not specified, it defaults to the value of <i>@Position</i> . <i>@AlternatePositions</i> SHALL be non-negative.
<i>ComponentRef</i> ?	IDREF	Specifies the Component that is to be loaded into this Feeder . This Component SHALL be an input of the Feeding Process.

Table 8-71: Feeder Element (Sheet 2 of 2)

Name	Data Type	Description
<i>FeederSynchronization</i> ?	enumeration	<p>Specifies the synchronization of multiple Feeder Elements with identical Component Elements:</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Alternate</i> – The feeders specified in <i>@Position</i> alternate. <i>Backup</i> – This Feeder is the backup feeder for the Component in case of a misfeed or malfunction. The priority of backup feeders is defined by their position in <i>@AlternatePositions</i>. <i>Chain</i> – This feeder is activated as soon as the feeder prior to it in the list is empty. <i>Primary</i> – This Feeder is the primary feeder for the Component.
<i>FeederType</i> ?	NMTOKEN	<p>Specifies the feeder type.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>AddOn</i> – Add on feeder (e.g., CDs). <i>BookBlock</i> – A feeder for book blocks. <i>Folding</i> – A folding feeder that folds the input Component or Media. <i>Gluing</i> – A gluing feeder <i>Roll</i> – Roll feeder for Web processes. These are also known as unwinders. <i>Sheet</i> – Single Sheet feeder. <i>Signature</i> – Single Signature feeder.
<i>Loading</i> ?	NMTOKEN	<p>Specifies the feeder loading.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Bundle</i> – Stream feeder, using the output of the Bundling Process. <i>FanFold</i> – Automatic loading of FanFold Media. <i>Manual</i> – Manual loading of stacks <i>Online</i> – Loaded by a gripper or conveyor. The "<i>Online</i>" value is also applicable for <i>@FeederType = "Roll"</i>. <i>PrintRoll</i> – Automatic loading of single products from a print Roll, using the output of the Winding Process.
<i>Opening</i> ?	enumeration	<p>Specifies the opening of Signatures:</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Back</i> – Overfold on back. <i>Front</i> – Overfold on front. <i>None</i> – Signatures are not opened. <i>Sucker</i> – Sucker opening, no overfold.
<i>Position</i> ?	integer	<p><i>@Position</i> of feeder on a collecting and gathering chain in chain movement direction. <i>@Position = "0"</i> is first feeder feeding to the collecting and gathering chain. Only one Feeder can be specified for any given <i>@Position</i>. If <i>@Position</i> is negative, it specifies the position counted from the back of the chain (e.g., "<i>-1</i>" = last position, "<i>-2</i>" = next to last position, etc.).</p>
<i>FeederQualityParams</i> ?	element	Definition of the setup and policy for feeding quality.

8.39.2 Element: FeederQualityParams

The FeederQualityParams Element defines the setup and policy for feeding quality control. It can be specified individually for each Feeder.

Table 8-72: FeederQualityParams Element

Name	Data Type	Description
<i>BadFeedQuality</i> ?	enumeration	Defines the operation of the bad feed quality control. Allowed values are from: @IncorrectComponentQuality.
<i>BadFeeds</i> ?	integer	Number of consecutive bad feeds until the Device stops.
<i>DoubleFeedQuality</i> ?	enumeration	Defines the operation of the double feed quality control. Allowed values are from: @IncorrectComponentQuality.
<i>DoubleFeeds</i> ?	integer	Number of consecutive double feeds until the Device stops.
<i>IncorrectComponentQuality</i> ?	enumeration	Defines the operation of the incorrect components quality control: Allowed values are: <i>NotActive</i> – Quality control is not active. <i>Check</i> – Check the quality and register. <i>Waste</i> – Check the quality and register. A component failing the test is waste. <i>StopNoWaste</i> – Check the quality and register. Device will stop after the defined number of consecutive errors. The error will be corrected (e.g., manually). <i>StopWaste</i> – Check the quality and register. A component failing the test is waste, and the Device will stop after the defined number of consecutive errors.
<i>IncorrectComponents</i> ?	integer	Number of consecutive incorrect components until the Device stops.

8.39.3 Element: CollatingItem

Table 8-73: CollatingItem Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Amount</i> ?	integer	Determines, how many consecutive items shall be consumed.
<i>BundleDepth</i> ?	integer	In case of nested bundles with @BundleType = "Stack", this parameter addresses the element to be consumed within the "tree" of such bundles. If the real bundle depth level (@BundleType = "Stack") is smaller than the value of @BundleDepth, individual stack items (i.e., the smallest available level) shall be consumed. If the input component referenced does not contain bundles, then this parameter is ignored. A @BundleDepth value of "0" means the Component itself. A value of "1" addresses the BundleItem Elements referenced from the Component (i.e., the Component/Bundle/BundleItem/Component(Ref), and so on).

Table 8-73: CollatingItem Element (Sheet 2 of 2)

Name	Data Type	Description
ComponentRef?	IDREF	References one of the input components to the Process to be (partially) consumed by the CollatingItem Element. This Component SHALL be an input of the Feeding Process.
Orientation?	Orientation	Named <i>@Orientation</i> of the CollatingItem relative to the input coordinate system. For details see Table 2-4, “Matrices and Orientation values for describing the orientation of a Component” on page 39. At most one of <i>@Orientation</i> or <i>@Transformation</i> SHALL be specified. If neither is specified, no transformation is applied. The transformation specified here is applied in addition to orientation/transformation specified in the respective Resource .
Transformation?	matrix	Orientation of the Component respective to the input coordinate system. This <i>@Transformation</i> specified here is applied in addition to orientation/transformation specified in the respective Resource . At most one of <i>@Orientation</i> and <i>@Transformation</i> SHALL be specified. If neither is specified, no transformation is applied.
TransformationContext?	enumeration	This parameter specifies the object, which is to be manipulated in orientation/transformation, and it is important to determine the sequence of stack items after flipping. Allowed values are: <i>StackItem</i> – Apply individually to the smallest element on the stack which can be manipulated individually (e.g., to a single Sheet in the case of a stack of Sheets). <i>Component</i> – Apply to each single element of a CollatingItem individually. <i>CollateItem</i> – apply to a CollatingItem as a whole. Note: If <i>@Amount = "1"</i> , Component and CollatingItem are referring to the same object and, therefore, result in the same output.

Note: Most real world Devices process stack items one by one, and hence will hardly ever support *@TransformationContext = "CollateItem"*. This requires some kind of buffer for the stack items belonging to a single collating item plus a flipping mechanism for **Winding** Process.

8.40 FileSpec

Specification of a file or a set of files. If a single **FileSpec** instance specifies a set of files, it SHALL do so using the *@FileFormat* and *@FileTemplate* Attributes to specify a sequence of URLs. Otherwise, each **FileSpec** instance specifies a single file. If that single file is inside a container file (e.g., a Zip file or is compressed or encoded as indicated by *@Compression*), the **FileSpec** instance SHALL define a *@ContainerRef* Attribute which defines another **FileSpec** instance that specifies the container file. In such a case, the Attributes of each **FileSpec** instance SHALL apply only to the properties of the file in the context of the container.

Resource Properties

Resource referenced by: **ApprovalDetails**, **AssetListCreationParams**, **ByteMap**, **Color**, **Color/**, **ColorCorrectionParams**, **ColorCorrectionParams/**, **ColorCorrectionOp**, **ColorSpaceConversionOp**, **ColorSpaceConversionParams**, **Device/IconList/Icon**, **DieLayout**, **ExposedMedia**, **FileSpec/@ContainerRef**, **FormatConversionParams/TIFFFormatParams/TIFFEmbeddedFile**, **LayoutElementProductionParams**, **PrintCondition**,

**QualityControlResult, RunList, ShapeDef,
ShapeDefProductionParams/ObjectModel,
ShapeDefProductionParams/ShapeTemplate**

Input of Processes: —

Output of Processes: —

Table 8-74: FileSpec Resource (Sheet 1 of 3)

Name	Data Type	Description
<i>CheckSum</i> ?	hexBinary	Checksum of the file being referenced using the RSA MD5 algorithm. In JDF 1.1A, the term RSA MD was completed to RSA MD5. The data type was chosen as hexBinary to accommodate the 128 bit output of the MD5 algorithm. The <i>@CheckSum</i> SHALL be calculated from the entire file, not just parts of the file.
<i>ContainerRef</i> ?	IDREF	References the container FileSpec for this file. When a container FileSpec is pointed to by <i>@ContainerRef</i> , that FileSpec SHALL NOT specify <i>@FileFormat</i> and <i>@FileTemplate</i> Attributes. The container mechanism MAY be used recursively (e.g., for a Zip file held in a tar file, a Zip file in a Zip file, an encoded Zip file, etc.). See Appendix K, “Resolving RunList/@Directory and FileSpec/@URL URI References” on page 1255 for details.
<i>Encoding</i> ?	NMTOKEN	Encoding or code page of the file contents. Values include those from: http://www.iana.org/assignments/character-sets .
<i>FileFormat</i> ?	string	A formatting string used with the <i>@FileTemplate</i> Attribute to define a sequence of URLs in a batch Process, each of which has the same semantics as the <i>@URL</i> Attribute. Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203. Constraint: if neither <i>@URL</i> nor <i>@UID</i> is present, both <i>@FileFormat</i> and <i>@FileTemplate</i> SHALL be present, unless the Resource is a pipe. If either <i>@URL</i> or <i>@UID</i> is specified, then <i>@FileFormat</i> and <i>@FileTemplate</i> SHALL NOT be specified.
<i>FileSize</i> ?	LongInteger	Size of the file in bytes.
<i>FileTemplate</i> ?	string	A template, used with <i>@FileFormat</i> , to define a sequence of URLs in a batch Process, each of which has the same semantics as the <i>@URL</i> Attribute. Constraint: if neither <i>@URL</i> nor <i>@UID</i> is present, both <i>@FileFormat</i> and <i>@FileTemplate</i> SHALL be present, unless the Resource is a pipe. Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.

Table 8-74: FileSpec Resource (Sheet 2 of 3)

Name	Data Type	Description
<i>MimeType</i> ?	string	<p>MIME type or file type of the file (or files of identical type when specifying a sequence of file names using the <i>@FileFormat</i> and <i>@FileTemplate</i> Attributes). See <i>@Compression</i> for the indication of compression or encoding of the file. See <i>@MimeTypeVersion</i> for the format version.</p> <p>If the file format has a MIME Media Type [iana-mt] registered with IANA, that value SHALL be used. The [RFC2046] defines that MIME Media Types are case-insensitive.</p> <p>If the file format does not have a MIME Media Type registered with IANA, then the XJDF spec defines string values, called file types, which SHALL be used.</p> <p>Values include those from: Appendix H, “MimeType and MimeTypeVersion Attributes” on page 1197.</p>
<i>MimeTypeVersion</i> ?	string	<p>The level or version of the file format identified by <i>@MimeType</i>, whether the value of <i>@MimeType</i> is a MIME Media Type or a file type value defined by the XJDF spec. Example values include:</p> <p><i>"PDF/1.3"</i>, <i>"PDF/1.4"</i> and <i>"PDF/X-1a:2001"</i> for <i>@MimeType = "application/pdf"</i></p> <p><i>"TIFF-IT/FP:1998"</i>, <i>"TIFF-IT/CT:1998"</i> and <i>"TIFF-IT/LW/P1:1998"</i> for <i>@MimeType = "TIFF/IT"</i></p> <p>Values include those from: Appendix H, “MimeType and MimeTypeVersion Attributes” on page 1197.</p>
<i>OverwritePolicy</i> ?	enumeration	<p>Policy that specifies the policy to follow when a file already exists and the FileSpec is used as an Output Resource.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Overwrite</i> – Overwrite the old file. <i>RenameNew</i> – Rename the new file. <i>RenameOld</i> – Rename the old file. <i>NewVersion</i> – Create a new file version. Only valid when the FileSpec references a file on a version aware file system. <i>OperatorIntervention</i> – Present a dialog to an operator. <i>Abort</i> – Abort the Process without modifying the old file.
<i>Password</i> ?	string	<p>Password or decryption key that is needed to read the file contents. Note: since this password string is not encrypted, it SHOULD only be passed around within a protected environment.</p>
<i>ResourceUsage</i> ?	NMTOKEN	<p>If an Element uses more than one FileSpec Subelement, this Attribute is used to refer from the parent Element to a certain child Element of this type, for example, see FormatConversionParams.</p> <p>Values include those from: Table 8-104, “ResourceUsage Attribute Values” on page 668.</p>
<i>SearchDepth</i> ?	integer	<p>Used when FileSpec refers to a directory to specify the maximum directory depth that will be recursively searched. 0 specifies this directory only, -1 specifies an unlimited search.</p>

Table 8-74: FileSpec Resource (Sheet 3 of 3)

Name	Data Type	Description
<i>UID</i> ?	NMTOKEN	<p>Internal ID of the referenced file. The <i>@UID</i> SHALL be unique within the workflow. This Attribute is dependent on the type of file that is referenced:</p> <p>Values include:</p> <ul style="list-style-type: none"> PDF – Variable unique identifier in the ID field of the PDF file's trailer. ICC Profile – The Profile ID in bytes 84-99 of the ICC profile header. Others – Format specific. <p>Constraint: If neither <i>@URL</i> nor <i>@UID</i> is present on an input FileSpec, and neither <i>@FileFormat</i> nor <i>@FileTemplate</i> is present, the referencing Resource SHALL be a pipe. If either <i>@URL</i> or <i>@UID</i> is specified, then <i>@FileFormat</i> and <i>@FileTemplate</i> SHALL NOT be specified.</p>
<i>URL</i> ?	URL	<p>Location of the file specified as either an Absolute URI or a Relative URI. If neither <i>@URL</i> nor <i>@UID</i> is present on an input FileSpec, and neither <i>@FileFormat</i> nor <i>@FileTemplate</i> is present, the referencing Resource SHALL be a pipe. If either <i>@URL</i> or <i>@UID</i> is specified, then <i>@FileFormat</i> and <i>@FileTemplate</i> SHALL NOT be specified.</p> <p>If <i>@URL</i> is not specified in an Output Resource, the system-specified location will be assumed, but this value SHALL be updated as soon as the Output Resource is available. For example, an instruction for a digital delivery XJDF Device to compress the files MAY specify the output RunList with the <i>@Compression</i> Attribute without the <i>@URL</i> Attribute.</p> <p>See [RFC3986] and Appendix K, “Resolving RunList/@Directory and FileSpec/@URL URI References” on page 1255 and Appendix N, “FileSpec Attributes and Container Subelement” on page 1281 for the syntax and examples. For the “<i>file</i>” URL scheme see also [RFC1738] and [FileURL].</p>
<i>UserFileName</i> ?	string	A user-friendly name which can be used to identify the file. MAY be used by a Controller to identify a file on a Device without knowing the file's internal location.
<i>Disposition</i> ?	element	Indicates what the Device SHOULD do with the file when the Process that uses this Resource completes. If not specified here or in the parent RunList , the file specified by this FileSpec SHOULD NOT be deleted by the Device.

— Attribute: ResourceUsage

Table 8-75: ResourceUsage Attribute Values (Sheet 1 of 2)

Value	Description
<i>AbstractProfile</i>	Used for ColorCorrectionOp/ FileSpec and ColorSpaceConversionOp/ FileSpec
<i>ColorProfile</i>	Used in Color/FileSpec

Table 8-75: ResourceUsage Attribute Values (Sheet 2 of 2)

Value	Description
<i>DeviceLinkProfile</i>	Used for ColorCorrectionOp/FileSpec and ColorSpaceConversionOp/FileSpec .
<i>FinalTargetDevice</i>	Used for ColorCorrectionParams/FileSpec and ColorSpaceConversionParams/FileSpec
<i>InputFormat</i>	Used for FormatConversionParams/FileSpec
<i>OutputFormat</i>	Used for FormatConversionParams/FileSpec
<i>OutputProfile</i>	Used for ExposedMedia/FileSpec
<i>SearchPath</i>	Used for AssetListCreationParams/FileSpec .
<i>SourceProfile</i>	Used for ColorSpaceConversionOp/FileSpec
<i>TargetProfile</i>	Used for PrintCondition/FileSpec and Color/FileSpec .
<i>WorkingColorSpace</i>	Used for ColorCorrectionParams/FileSpec and ColorSpaceConversionParams/FileSpec

8.40.1 Element: Disposition

This Element describes how long an asset SHOULD be maintained by a Device. The Device will perform an action defined by *Disposition/@DispositionAction* when a “disposition time” occurs. Disposition time is defined either as:

$$\begin{aligned} Until &\leq "Disposition \text{ time}" \leq Until + ExtraDuration \\ ProcessCompleteTime + MinDuration &\leq "Disposition \text{ time}" \leq \\ &ProcessCompleteTime + MinDuration + ExtraDuration \end{aligned}$$
Table 8-76: Disposition Element (Sheet 1 of 2)

Name	Data Type	Description
<i>DispositionAction</i> ?	enumeration	Allowed values are: <i>Delete</i> – The asset is deleted when disposition time occurs. <i>Archive</i> – The asset is archived when disposition time occurs.
<i>DispositionUsage</i> ?	enumeration	Default behavior: Disposition applies to all Processes that link to the Disposition Resource (if <i>@DispositionUsage</i> not specified). Allowed values are: <i>Input</i> – Disposition applies only to Processes that use the asset as an Input Resource. <i>Output</i> – Disposition applies only to Processes that use the asset as an Output Resource.
<i>ExtraDuration</i> ?	duration	Indicates the maximum duration that the Device is allowed to retain the asset after the time specified by <i>@MinDuration</i> or <i>@Until</i> . If <i>@ExtraDuration</i> , <i>@MinDuration</i> and <i>@Until</i> are all unspecified, the asset is retained for a system specified time.
<i>MinDuration</i> ?	duration	Indicates the minimum duration that the Device SHOULD retain the asset after the Process that uses the asset completes.

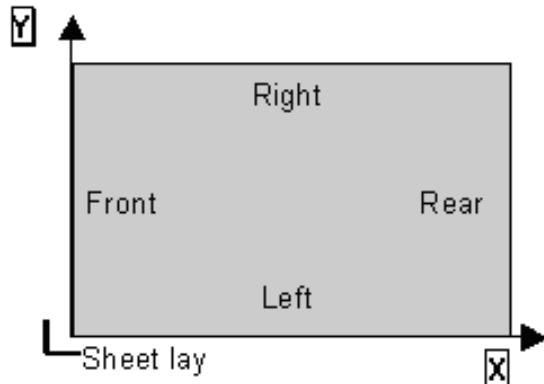
Table 8-76: Disposition Element (Sheet 2 of 2)

Name	Data Type	Description
Priority?	integer	Value between 0 and 100 that specifies the order in which assets are deleted or archived when the values of <i>@ExtraDuration</i> , <i>@MinDuration</i> and <i>@Until</i> cannot be honored (e.g., when local storage runs low). Assets with <i>@Priority = "0"</i> will be deleted first.
Until?	dateTime	Indicates an absolute point in time when the Device or application SHOULD stop the asset retention. If <i>@Until</i> is specified, <i>@MinDuration</i> SHALL be ignored.

8.41 FoldingParams

This Resource describes the folding parameters, including the sequence of folding steps. It is also possible to execute the predefined steps of the folding catalog. After each folding step of a folding procedure, the origin of the coordinate system is moved to the lower left corner of the intermediate folding product. For details see Section 2.5.4, “Product Example: Simple Brochure” on page 40.

The specification of reference edges (i.e., “Front”, “Rear”, “Left” and “Right”) for the description of an operation (e.g., the positioning of a tool) is done by means of determined names as shown in Figure 8-33, below.

Figure 8-27: Names of the reference edges of a Sheet in the FoldingParams Resource

Resource Properties

Resource referenced by: —

Intent Pairing: **FoldingIntent**

Input of Processes: **Folding**

Output of Processes: —

Table 8-77: FoldingParams Resource

Name	Data Type	Description
<i>FoldCatalog</i> ?	NMTOKEN	Describes the type of fold according to the folding catalog in Figure 8-34, “Fold catalog part 1,” on page 674 and Figure 8-35, “Fold catalog part 2,” on page 675. In case of any ambiguity, the folding notation takes precedence over the graphic illustration in the aforementioned Figures. Value format is: " <i>Fn-i</i> " where "n" is the number of finished pages and "i" is either an integer, which identifies a particular fold or the letter "X", which identifies a generic fold (e.g., " <i>F6-2</i> " describes a Z-fold of 6 finished pages, and " <i>F6-X</i> " describes a generic fold with 6 finished pages).
<i>FoldingDetails</i> ?	NMTOKEN	<i>@FoldingDetails</i> is a system dependent descriptor of the folding. <i>@FoldingDetails</i> MAY be used to differentiate differing fold dimensions with the same general topology.
<i>SheetLay</i> ?	enumeration	Lay of input media. Allowed values are: <i>Left</i> <i>Right</i> Note: <i>@SheetLay</i> does not modify the coordinate references of the Folding Process.
<i>Crease</i> *	element	Defines one or more <i>Crease</i> lines.
<i>Cut</i> *	element	<i>Cut</i> Elements describe an individual cut.
<i>FileSpec</i> (CIP3) ?	element	Reference to a CIP3 file that contains folding instructions in the CIP3 format.
<i>Fold</i> *	element	Describes the folding operations in the sequence in which they are to be carried out. If both <i>@FoldCatalog</i> and <i>Fold</i> Elements are specified, the <i>Fold</i> Elements have precedence, and the <i>@FoldCatalog</i> specifies only the topology. For instance a cover-fold with a page size ratio of 0.52 to 0.48 would still be defined as an " <i>F4-1</i> ".
<i>Perforate</i> *	element	Defines one or more <i>Perforate</i> lines. The order of <i>Cut</i> , <i>Crease</i> , <i>Fold</i> , <i>Perforate</i> specifies the sequence of operations in the folding machine.

Figure 8-28: Fold catalog part 1

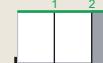
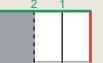
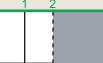
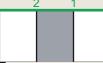
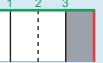
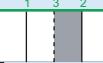
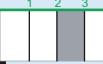
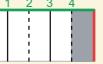
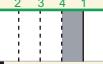
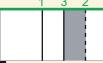
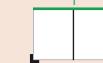
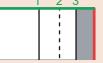
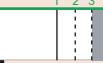
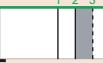
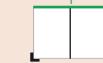
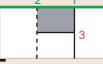
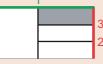
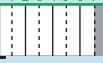
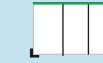
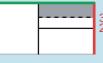
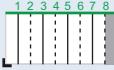
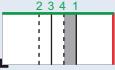
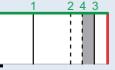
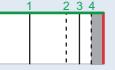
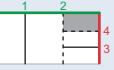
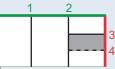
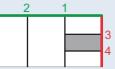
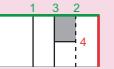
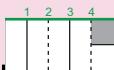
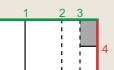
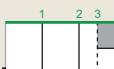
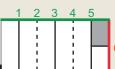
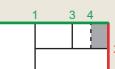
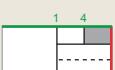
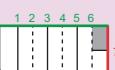
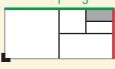
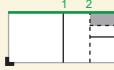
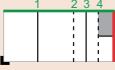
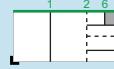
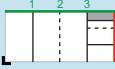
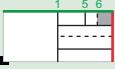
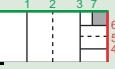
F2-1 	F4-1 2x1 	F4-2 2x1 	F6-1 3x1 	F6-2 3x1 
F6-3 3x1 	F6-4 3x1 	F6-5 3x1 	F6-6 3x1 	F6-7 3x1 
F6-8 3x1 	F8-1 4x1 	F8-2 4x1 	F8-3 4x1 	F8-4 4x1 
F8-5 4x1 	F8-6 4x1 	F8-7 2x2 	F10-1 5x1 	F10-2 5x1 
F10-3 5x1 	F12-1 6x1 	F12-2 6x1 	F12-3 6x1 	F12-4 6x1 
F12-5 6x1 	F12-6 6x1 	F12-7 3x2 	F12-8 3x2 	F12-9 3x2 
F12-10 3x2 	F12-11 3x2 	F12-12 2x3 	F12-13 2x3 	F12-14 2x3 
F14-1 7x1 	F16-1 8x1 	F16-2 8x1 	F16-3 8x1 	F16-4 8x1 
F16-5 8x1 	F16-6 4x2 	F16-7 4x2 	F16-8 4x2 	F16-9 4x2 
F16-10 4x2 	F16-11 4x2 	F16-12 4x2 	F16-13 2x4 	F16-14 2x4 

Figure 8-29: Fold catalog part 2

F18-1 9x1	F18-2 9x1	F18-3 9x1	F18-4 9x1	F18-5 3x3
 ↑1/9 ↓1/9 ↑1/9 ↓1/9 ↑1/9 ↓1/9 ↑1/9 ↓1/9	 ↑2/3 ↓1/3 ↑1/9 ↓1/9	 ↑1/3 ↓1/3 ↑2/9 ↓1/9	 ↑1/3 ↓1/3 ↑1/9 ↓1/9	 ↑1/3 ↓1/3 + ↑1/3 ↓1/3
F18-6 3x3	F18-7 3x3	F18-8 3x3	F18-9 3x3	F20-1 5x2
 ↑1/3 ↓1/3 + ↑2/3 ↓1/3	 ↑1/3 ↑1/3 + ↑1/3 ↓1/3	 ↑1/3 ↑1/3 + ↑2/3 ↓1/3	 ↑2/3 ↑1/3 + ↑2/3 ↑1/3	 ↑2/5 ↓2/5 ↑1/5 + ↑1/2
F20-2 5x2	F24-1 6x2	F24-2 6x2	F24-3 6x2	F24-4 6x2
 ↑1/5 ↓1/5 ↑1/5 ↓1/5 + ↑1/2	 ↑1/3 ↓1/3 + ↑1/2 + ↑1/6	 ↑1/3 ↑1/3 + ↑1/2 + ↑1/6	 ↑1/3 ↓1/3 ↑1/6 + ↑1/2	 ↑1/3 ↓1/3 ↓1/6 + ↑1/2
F24-5 6x2	F24-6 6x2	F24-7 6x2	F24-8 3x4	F24-9 3x4
 ↑1/3 ↑1/3 ↓1/6 + ↑1/2	 ↑1/6 ↓1/6 ↑1/6 ↓1/6 + ↑1/2	 ↑1/3 + ↑1/2 + ↑1/3 ↓1/6	 ↑1/3 ↓1/3 + ↑1/2 ↓1/4	 ↑2/3 ↑1/3 + ↑1/2 ↓1/4
F24-10 3x4	F24-11 4x3	F28-1 7x2	F32-1 16x1	F32-2 8x2
 ↑1/3 ↑1/3 + ↑1/2 ↓1/4	 ↑1/2 + ↑2/3 ↓1/3 + ↑1/4	 ↑1/7 ↓1/7 ↑1/7 ↓1/7 ↑1/7 ↓1/7 + ↑1/2	 ↑1/2 ↓1/4 ↑1/8 ↓1/16	 ↑1/2 ↓1/4 + ↑1/2 + ↑1/8
F32-3 8x2	F32-4 4x4	F32-5 4x4	F32-6 4x4	F32-7 4x4
 ↑1/2 ↓1/4 + ↑1/2 ↓1/8	 ↑1/2 + ↑1/2 + ↑1/4 + ↑1/4	 ↑1/2 + ↑1/2 + ↓1/4 + ↓1/4	 ↑1/2 + ↑1/2 + ↑1/4 + ↓1/4	 ↑1/4 ↓1/4 ↑1/4 + ↑1/2 ↓1/4
F32-8 4x4	F32-9 4x4	F36-1 9x2	F36-2 6x3	F40-1 5x4
 ↑1/2 ↓1/4 + ↑1/2 ↓1/4	 ↑1/2 + ↑1/2 ↓1/4 + ↑1/4	 ↑1/3 ↓1/3 ↑1/9 ↓1/9 + ↑1/2	 ↑1/3 ↓1/3 + ↑1/3 ↓1/3 + ↑1/6	 ↑1/5 ↓1/5 ↑1/5 ↓1/5 + ↑1/2 ↓1/4
F48-1 6x4	F48-2 4x6	F64-1 8x4	F64-2 8x4	
 ↑1/3 ↓1/3 + ↑1/4 ↓1/4 ↑1/4 + ↑1/6	 ↑1/4 ↓1/4 ↑1/4 + ↑1/3 ↓1/3 ↑1/6	 ↑1/2 + ↑1/4 ↓1/4 ↑1/4 + ↑1/4 ↓1/8	 ↑1/4 ↓1/4 ↑1/4 + ↑1/4 ↓1/4 ↑1/4 + ↑1/6	

LEGEND

- Fold up
- - - Fold down
- Finished format folded sheet
- 1, 2, 3 Folds in numeric order
- L Lay

Example: F32-3, 8x2

F32-3: Signature with 32 pages

8x2: Split: 8 sheet parts lengthwise 2 sheet parts cross

↑1/2: Fold up with 1/2 of the open sheet format length

↓1/4: Fold down with 1/4 of the open sheet format length

+ : Fold direction change: 90...

↑1/2: Fold up with 1/2 of the open sheet format

+ : Fold direction change: 90...

↓1/8: Fold down with 1/8 of the open sheet format length

8.42 FontPolicy

This Resource defines the policies that Devices follow when font errors occur while PDL files are being processed. When fonts are referenced by PDL files but are not provided, Devices SHALL provide one of the following two fallback behaviors:

- 1 The Device provides a standard default font which is substituted whenever a font cannot be found.
- 2 The Device provides an emulation of the missing font.

If neither fallback behavior is requested (i.e., both `@UseDefaultFont` and `@UseFontEmulation` are `"false"`), then the Job will fail if a referenced font is not provided. The **FontPolicy** allows Jobs to specify whether either of these fallback behaviors are to be employed when missing fonts occur.

Resource Properties

Resource referenced by: —

Input of Processes: *Interpreting, Trapping*

Output of Processes: —

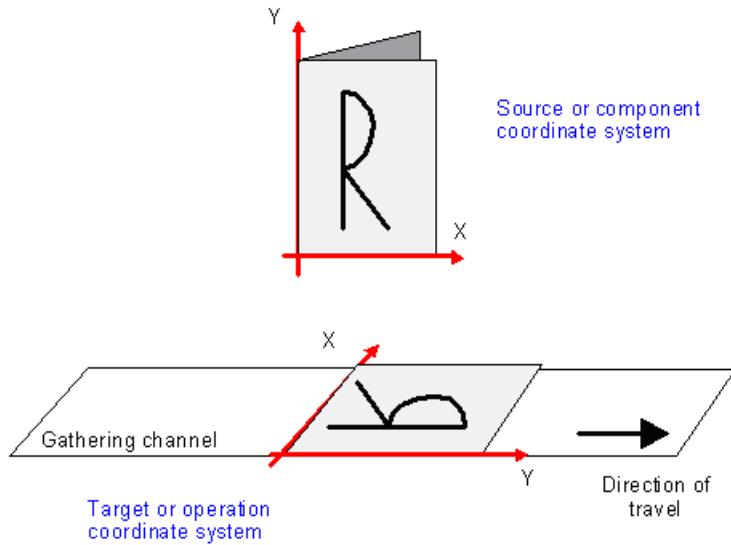
Table 8-78: FontPolicy Resource

Name	Data Type	Description
<code>PreferredFont</code>	NMTOKEN	The name of a font to be used as the default font for this Job. It is not an error if the Device cannot use the specified font as its default font.
<code>UseDefaultFont</code>	boolean	If <code>"true"</code> , the Device SHALL resort to a default font if a font cannot be found. This is the normal behavior of the PostScript interpreter, which defaults to Courier when a font cannot be found.
<code>UseFontEmulation</code>	boolean	If <code>"true"</code> , the Device SHALL emulate a requested font if a font cannot be found.

8.43 FormatConversionParams

8.44 GatheringParams

This Resource contains the Attributes of the **Gathering** Process.

Figure 8-30: Coordinate system used for Gathering**Resource Properties**

Resource referenced by: —

Input of Processes: **Gathering**

Output of Processes: —

Table 8-79: GatheringParams Resource

Name	Data Type	Description
Disjoining ?	element	Description of the separation properties between individual components on a gathered pile.

Resource**8.45 GluingParams****GluingParams** define the parameters applying a generic line of glue to a component.**Resource Properties**

Resource referenced by: —

Intent Pairing: **BindingIntent**Input of Processes: **Gluing**

Output of Processes: —

Table 8-80: GluingParams Resource

Name	Data Type	Description
<i>GluingProductionID</i> ?	NMTOKEN	Defines a gluing scheme for production.
<i>GlueLine</i> *	element	Definition of one or more <i>GlueLine</i> line applications.

8.46 HeadBandApplicationParams

This Resource specifies how to apply headbands in hard cover book production.

Resource Properties

Resource referenced by: —
 Input of Processes: *HeadBandApplication*
 Output of Processes: —

Table 8-81: HeadBandApplicationParams Resource

Name	Data Type	Description
<i>BottomColor</i> ?	NamedColor	Color of the bottom head band.
<i>BottomColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If <i>@BottomColorDetails</i> is supplied, <i>@BottomColor</i> SHOULD also be supplied.
<i>BottomLength</i> ?	double	Length of the carrier material of the bottom head band along binding edge. If not specified, both head bands are on one carrier.
<i>BottomRef</i> ?	IDREF	Reference to a MiscConsumable that represents the bottom Head-Band.
<i>StripMaterial</i> ?	enumeration	Strip material. Allowed values are: <i>Calico</i> <i>Cardboard</i> <i>CrepePaper</i> <i>Gauze</i> <i>Paper</i> <i>PaperlinedMules</i> <i>Tape</i>
<i>TopColor</i> ?	NamedColor	Color of the top head band.
<i>TopColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If <i>@TopColorDetails</i> is supplied, <i>@TopColor</i> SHOULD also be supplied.
<i>TopLength</i> ?	double	Length of carrier material of the top head band along binding edge. If not specified, both head bands are on one carrier which has the length of the book block.
<i>TopRef</i> ?	IDREF	Reference to a MiscConsumable that represents the top HeadBand.
<i>Width</i> ?	double	Width of the head bands and carrier.
<i>GlueLine</i> *	element	The carrier can be applied to the book block with glue. The coordinate system for the <i>GlueLine</i> is defined in the Section 8.57, “EndSheetGluingParams”.

8.47 HoleMakingParams

This Resource specifies where to make a hole of what shape in components. This information is used by the **HoleMaking** Process.

Resource Properties

Resource referenced by: *LooseBindingParams*,
 Intent Pairing: *HoleMakingIntent*
 Input of Processes: *HoleMaking*

Output of Processes: —**Table 8-82: HoleMakingParams Resource**

Name	Data Type	Description
HolePattern *	element	Description of individual or lines of HolePattern Elements.

8.48 IdentificationField

This Resource contains information about a mark on a document (e.g., a bar code) used for OCR-based verification purposes or document separation.

Resource Properties

Resource referenced by:	Resource, Disjoining, EmbossingParams/Emboss , Layout/MarkObject , Content/BarcodeProductionParams
Input of Processes:	Verification
Output of Processes:	—

Table 8-83: IdentificationField Resource (Sheet 1 of 3)

Name	Data Type	Description
<i>BoundingBox</i> ?	rectangle	<p>Box that provides the boundaries of the mark that indicates where the IdentificationField is placed. If the IdentificationField is specified in a Layout, the coordinate system is defined by the MarkObject containing the IdentificationField. If no Layout context is available, the origin of the coordinate system is defined as the lower left corner of the Resource surface that <i>@Position</i> specifies when the specified surface is viewed in its natural orientation.</p> <p>Each item in the list below specifies a value of <i>@Position</i> and the corner that is the origin for the specified value when the viewer is positioned in front of the front surface. For example, when <i>@Position</i> = "Left", the origin is the bottom-back corner of left surface when viewed from the front surface of the Resource and lower-left corner when viewed from the left surface.</p> <ul style="list-style-type: none"> • "Front" – Bottom-left corner • "Left" – Bottom-back corner • "Back" – Bottom-right corner • "Right" – Bottom-front corner • "Top" – Front-left corner • "Bottom" – Back left corner <p>If no <i>@BoundingBox</i> is defined and the IdentificationField is specified</p> <ul style="list-style-type: none"> • outside the context of a Layout, the complete visible surface SHALL be scanned for an appropriate bar code. • within the context of a Layout, the implied <i>@BoundingBox</i> is specified by MarkObject/@ClipBox. <p>The <i>@BoundingBox</i> is used only as metadata when searching or scanning IdentificationField Elements and not used when generating IdentificationField Elements in a LayoutElementProduction Process.</p>

Table 8-83: IdentificationField Resource (Sheet 2 of 3)

Name	Data Type	Description
<i>Encoding</i> ?	enumeration	Encoding of the information. Allowed values are: <i>ASCII</i> – Plain-text font. <i>Barcode</i> – Any bar code. <i>Braille</i> – Braille text. <i>RFID</i> – Radio Frequency Identification tag.
<i>EncodingDetails</i> ?	NMTOKEN	Details about the encoding type. An example is the bar code scheme. Values include those from: Table 8-119, “EncodingDetails Attribute Values”.
<i>Format</i> ?	regExp	Regular expression that defines the expected format of the expression (e.g., the number of digits, alphanumeric or numeric). Note that this field MAY also be used to define constant fields (e.g., the end of document markers or packaging labels). If not specified, any expression is valid. Exactly one of <i>@Format</i> , <i>@Value</i> or the pair <i>@ValueFormat</i> and <i>@ValueTemplate</i> SHALL be specified.
<i>Orientation</i> ?	matrix	Orientation of the contents within the IdentificationField . The coordinate system is defined in the system of the Sheet or component where the IdentificationField resides. The <i>@Orientation</i> is used only as metadata when searching or scanning IdentificationField Elements and not used when generating IdentificationField Elements in a LayoutElementProduction Process.
<i>Page</i> ?	integer	If <i>@Position</i> = "Page", this refers to the page where the IdentificationField can be found. Negative values denote an offset relative to the last page in a stack of pages.
<i>Position</i> ?	enumeration	Position with respect to the Instance Document or Resource to which the Resource refers. Allowed values are: <i>Header</i> – Sheet before the document. <i>Trailer</i> – Sheet after the document. <i>Page</i> – A page of the document. <i>Top</i> – The top of the Resource. <i>Bottom</i> – The bottom of the Resource. <i>Left</i> – The left side of the Resource. <i>Right</i> – The right side of the Resource. <i>Front</i> – The front side of the Resource. <i>Back</i> – The back side of the Resource.
<i>Purpose</i> ?	enumeration	Purpose defines the usage of the field. Allowed values are: <i>Label</i> – Used to mark a product or component. <i>Separation</i> – Used to separate documents. <i>Verification</i> – Used for verification of documents.

Table 8-83: IdentificationField Resource (Sheet 3 of 3)

Name	Data Type	Description
<i>PurposeDetails</i> ?	NMTOKEN	More detail about the usage of the barcode. Values include: <i>ProductIdentification</i> – End product identification (e.g., scanning in the super market).
<i>Value</i> ?	string	Fixed value of the IdentificationField (e.g., on a label). Exactly one of <i>@Format</i> , <i>@Value</i> or the pair <i>@ValueFormat</i> and <i>@ValueTemplate</i> SHALL be specified.
<i>ValueFormat</i> ?	string	A formatting string used with <i>@ValueTemplate</i> to define fixed and/or variable content of barcodes or text. Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203. Constraint: exactly one of <i>@Format</i> , <i>@Value</i> or the pair <i>@ValueFormat</i> and <i>@ValueTemplate</i> SHALL be specified.
<i>ValueTemplate</i> ?	string	A list of values used with <i>@ValueFormat</i> to define fixed and/or variable content of barcodes or text. If MetadataMap Elements are present, MetadataMap/@Name SHALL be included in <i>@ValueTemplate</i> to select the data from the MetadataMap . Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203. Constraint: exactly one of <i>@Format</i> , <i>@Value</i> or the pair <i>@ValueFormat</i> and <i>@ValueTemplate</i> SHALL be specified.
<i>BarcodeDetails</i> ?	element	Additional specification for complex barcodes.
<i>ExtraValues</i> ?	element	Additional values encoded in the IdentificationField .
<i>MetadataMap</i> *	element	Describes the mapping of metadata that is encoded in an IdentificationField to <i>@PartIDKeys</i> . This allows for automated selective finishing based on bar codes.

— Attribute: EncodingDetails

The following list provides a sample of Barcode encoding details. Values that are not present in this list MAY be valid in an **XJDF** workflow.

Table 8-84: EncodingDetails Attribute Values (Sheet 1 of 2)

Value	Description	Value	Description
<i>BOBST</i>		<i>ITF_14</i>	
<i>BrailleASCII</i>	A binary representation for 6 dot Braille messages. See http://en.wikipedia.org/wiki/Braille_ASCII	<i>ITF_6</i>	
<i>BrailleUnicode</i>	A binary representation for Braille messages. See http://www.unicode.org/charts/PDF/U2800.pdf#search=%22braille%20unicode%22	<i>ITF_16</i>	

Table 8-84: EncodingDetails Attribute Values (Sheet 2 of 2)

Value	Description	Value	Description
<i>CODABAR</i>		<i>KURANDT</i>	
<i>CODABAR_Tradional</i>		<i>LAETUS_PHARMA</i>	
<i>CODABLOCK</i>		<i>MSI</i>	
<i>CODABLOCK_F</i>		<i>NDC_HRI</i>	
<i>Code128</i>		<i>PARAF</i>	
<i>Code25</i>		<i>Plessey</i>	
<i>Code39</i>		<i>PDF417</i>	
<i>Code39_Extended</i>		<i>PZN</i>	
		<i>QR</i>	
<i>EAN</i>	includes Bookland_EAN and ISSN.	<i>RSS_14</i>	
<i>EAN_13</i>		<i>RSS_14_Stacked</i>	
<i>EAN_8</i>		<i>RSS_14_Stacked_Omnidir</i>	
<i>EAN_Coupon</i>		<i>RSS_14_Truncated</i>	
<i>EAN_128</i>		<i>RSS_Limited</i>	
<i>HIBC_Code39</i>		<i>RSS_Expanded</i>	
<i>HIBC_Code128</i>		<i>RSS_Expanded_Stacked</i>	
<i>HIBC_Code39_2</i>		<i>UPC_A</i>	
<i>HIBC_CODABLOCK_F</i>		<i>UPC_Coupon</i>	
<i>HIBC_QR</i>		<i>UPC_E</i>	
<i>HIBC_DATAMATRIX</i>		<i>UPC_SCS</i>	
<i>Interleave25</i>			

8.48.1 Element: BarcodeDetails

Table 8-85: BarcodeDetails Element (Sheet 1 of 2)

Name	Data Type	Description
<i>BarcodeVersion ?</i>	NMTOKEN	<p>The version of a barcode.</p> <p>Values include those from: Table 8-123, “BarcodeVersion Values – for HIBC_DATAMATRIX” on page 690.</p> <p>Values include those from: Table 8-124, “BarcodeVersion Values – for QR barcodes” on page 691.</p>

Table 8-85: BarcodeDetails Element (Sheet 2 of 2)

Name	Data Type	Description
<i>ErrorCorrectionLevel</i> ?	NMTOKEN	<p>Error correction level for barcodes having a separately definable error correction level.</p> <p>Each value can be used only for certain values of IdentificationField/@EncodingDetails.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>PDF417_EC_0</i> – for @EncodingDetails = "PDF417" <i>PDF417_EC_1</i> – for @EncodingDetails = "PDF417" <i>PDF417_EC_2</i> – for @EncodingDetails = "PDF417" <i>PDF417_EC_3</i> – for @EncodingDetails = "PDF417" <i>PDF417_EC_4</i> – for @EncodingDetails = "PDF417" <i>PDF417_EC_5</i> – for @EncodingDetails = "PDF417" <i>PDF417_EC_6</i> – for @EncodingDetails = "PDF417" <i>PDF417_EC_7</i> – for @EncodingDetails = "PDF417" <i>PDF417_EC_8</i> – for @EncodingDetails = "PDF417" <i>QR_EC_L</i> – for @EncodingDetails = "QR" <i>QR_EC_M</i> – for @EncodingDetails = "QR" <i>QR_EC_Q</i> – for @EncodingDetails = "QR" <i>QR_EC_H</i> – for @EncodingDetails = "QR"
<i>XCells</i> ?	integer	<p>The number of cells in x direction of a matrix barcode. For "DATAMATRIX" this field can be omitted since @BarcodeVersion already defines this.</p> <p>For "PDF417" this is the number of codewords/row.</p>
<i>YCells</i> ?	integer	<p>The number of cells in y direction of a matrix barcode For "DATAMATRIX" this field can be omitted since @BarcodeVersion already defines this.</p> <p>For "PDF417" this is the number of rows.</p>

8.48.2 Element: ExtraValues

Table 8-86: ExtraValues Element

Name	Data Type	Description
<i>Usage</i>	NMTOKEN	<p>The usage of the value.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Supplemental</i> – UPC supplemental 2/5 digit symbology <i>CompositeCode</i> – This is applicable for barcodes like RSS-14 that have an optional composite code part. <i>Coupon</i> – The additional message for the EAN128 part of a UPC or EAN coupon.
<i>Value</i>	string	Additional value of the IdentificationField as specified in @Usage .

8.48.3 Usage of barcode Attributes

The following table specifies whether the Attributes **@Height**, **@Magnification** and **@Ratio** are applicable for a given barcode type that is specified by **@EncodingDetails**.

Table 8-87: Usage of Barcode Attributes for Certain Barcode Types

EncodingDetails values (barcode types)	Height	Magnification	Ratio
<i>Code25</i> <i>Code39</i> <i>Code39_Extended</i> <i>Interleave25</i> <i>MSI</i> <i>Plessey</i>	Used	Used	Used
<i>CODABAR</i> <i>Code128</i> <i>EAN_13</i> <i>EAN_8</i> <i>EAN_128</i> <i>HIBC_Code39</i> <i>HIBC_Code128</i> <i>ITF_14</i> <i>ITF_16</i> <i>NDC_HRI</i> <i>PARAF</i> <i>UPC_A</i> <i>UPC_E</i> <i>UPC_SCS</i> <i>UPC_SCS</i>	Used	Used	Not used
<i>BOBST</i> <i>KURANDT</i> <i>LAETUS_PHARMA</i>	Used	Not used	Not used
<i>RSS_14</i> <i>RSS_14_Stacked</i> <i>RSS_14_Stacked_Omnidir</i> <i>RSS_14_Truncated</i> <i>RSS_Limited</i> <i>RSS_Expanded</i> <i>RSS_Expanded_Stacked</i>	Not used	Used	Not used
<i>PZN</i>	Not used	Not used	Not used

— Attribute: BarcodeVersion – for HIBC_DATAMATRIX

The following table specifies valid values of `BarcodeDetails/@BarcodeVersion` for a barcode:

Table 8-88: BarcodeVersion Values – for HIBC_DATAMATRIX (Sheet 1 of 2)

Values			
<i>DM_8_by_18</i>	<i>DM_16_by_16</i>	<i>DM_26_by_26</i>	<i>DM_72_by_72</i>
<i>DM_8_by_32</i>	<i>DM_16_by_36</i>	<i>DM_32_by_32</i>	<i>DM_80_by_80</i>
<i>DM_10_by_10</i>	<i>DM_16_by_48</i>	<i>DM_40_by_40</i>	<i>DM_88_by_88</i>

Table 8-88: BarcodeVersion Values – for HIBC_DATAMATRIX (Sheet 2 of 2)

Values			
DM_12_by_12	DM_18_by_18	DM_44_by_44	DM_96_by_96
DM_12_by_26	DM_20_by_20	DM_48_by_48	DM_104_by_104
DM_12_by_36	DM_22_by_22	DM_52_by_52	DM_120_by_120
DM_14_by_14	DM_24_by_24	DM_64_by_64	DM_132_by_132
			DM_144_by_144

— Attribute: BarcodeVersion – for QR barcodes

The following table specifies valid values of BarcodeDetails/@BarcodeVersion for a QR barcode.

Table 8-89: BarcodeVersion Values – for QR barcodes

Values							
QR_1	QR_6	QR_11	QR_16	QR_21	QR_26	QR_31	QR_36
QR_2	QR_7	QR_12	QR_17	QR_22	QR_27	QR_32	QR_37
QR_3	QR_8	QR_13	QR_18	QR_23	QR_28	QR_33	QR_38
QR_4	QR_9	QR_14	QR_19	QR_24	QR_29	QR_34	QR_39
QR_5	QR_10	QR_15	QR_20	QR_25	QR_30	QR_35	QR_40

Example 8-12: Barcode

The following example illustrates the description of a barcode in a *LayoutElementProduction* Process:

TBD 2.x Example.

```
<LayoutElementProductionParams Class="Parameter" ID="BarcodeParams"
    Status="Available">
    <LayoutElementPart>
        <BarcodeProductionParams>
            <IdentificationField Encoding="Barcode" EncodingDetails="EAN_13"
                Purpose="Label" PurposeDetails="ProductIdentification"
                Value="0123456789128"/>
            <BarcodeReproParams Height="73.50" Magnification="1.0">
                <BarcodeCompParams CompensationProcess="Printing"
                    CompensationValue="10.0"/>
            </BarcodeReproParams>
        </BarcodeProductionParams>
    </LayoutElementPart>
</LayoutElementProductionParams>
```

8.49 ImageCompressionParams

Prior to JDF 1.2 the filtering in ImageCompressionParams was based on the terminology in PostScript and PDF. Many image compression and decompression filters require additional information in the form of a filter parameter dictionary, and additional filter parameters have been added to meet this need.

Resource Properties

Resource referenced by: ContentFormatConversionParams, RunList

Input of Processes: PDLCreation

Output of Processes: —

Table 8-90: ImageCompressionParams Resource

Name	Data Type	Description
ImageCompression *	element	Specifies how images are to be compressed. The elements are applied in sequential order.

8.49.1 Element: ImageCompression

Table 8-91: ImageCompression Element (Sheet 1 of 3)

Name	Data Type	Description
<i>AntiAliasImages</i> ?	boolean	If "true", anti-aliasing is permitted on images. If "false", anti-aliasing is not permitted. Anti-aliasing increases the number of bits per component in downsampled images to preserve some of the information that is otherwise lost by downsampling. Anti-aliasing is only performed if the image is actually downsampled and if <i>@ImageDepth</i> has a value greater than the number of bits per color component in the input image.
<i>AutoFilterImages</i> ?	boolean	SHALL NOT be specified unless <i>@EncodeImages</i> is "true". This Attribute is not used if <i>@ImageType</i> = "Monochrome". If "true", the filter defined by <i>@ImageAutoFilterStrategy</i> is applied to photos and the "FlateEncode" filter is applied to screen shots. If "false", the <i>@ImageFilter</i> compression method is applied to all images.
<i>ConvertImagesToIndexed</i> ?	boolean	If "true", the application converts images that use fewer than 257 colors to an indexed color space for compactness. This Attribute is used only when <i>@ImageType</i> = "Color".
<i>DCTQuality</i> ?	double	A value between 0 and 1 that indicates "how much" the Process is to compress images when using a "DCTEncode" filter. 0.0 means "do as loss-less compression as possible." 1.0 means "do the maximum compression possible."
<i>DownsampleImages</i> ?	boolean	If "true", sampled color images are downsampled using the resolution specified by <i>@ImageResolution</i> . If "false", downsampling is not carried out and the image resolution in the PDF file is the same as that in the source file.
<i>EncodeImages</i> ?	boolean	If "true", images are encoded using the compression filter specified by the value of the <i>@ImageFilter</i> key. If "false", no compression filters are applied to sampled images.

Table 8-91: ImageCompression Element (Sheet 2 of 3)

Name	Data Type	Description
<i>ImageAutoFilterStrategy</i> ?	NMTOKEN	Selects what image compression strategy to employ if passing through an image that is not already compressed. Values include: <i>JPEG</i> – Lossy JPEG compression for low-frequency images and lossless Flate compression for high-frequency images. <i>JPEG2000</i> – Lossy JPEG2000 compression for low-frequency images and lossless JPEG2000 compression for high-frequency images.
<i>ImageDepth</i> ?	integer	Specifies the number of bits per component in the downsampled image when <i>@DownsampleImages</i> = "true". If not specified, the downsampled image has the same number of bits per sample as the original image.
<i>ImageDownsampleThreshold</i> ?	double	Sets the image downsample threshold for images. This is the ratio of image resolution to output resolution above which downsampling can be performed. The following short examples provide a hypothetical configuration: To use <i>@ImageDownsampleThreshold</i> , set the following Attributes to the values indicated: <i>@ImageResolution</i> = 72 <i>@ImageDownsampleThreshold</i> = 1.5 The input image would not be downsampled unless it has a resolution greater than $(72 * 1.5) = 108$ dpi
<i>ImageDownsampleType</i> ?	enumeration	Downsampling algorithm for images. Allowed values are: <i>Average</i> – The program averages groups of samples to get the new downsampled value. <i>Bicubic</i> – The program uses bicubic interpolation on a group of samples to get a new downsampled value. <i>Subsample</i> – The program picks the middle sample from a group of samples to get the new downsampled value.

Table 8-91: ImageCompression Element (Sheet 3 of 3)

Name	Data Type	Description
<i>ImageFilter</i> ?	NMOKEN	<p>Specifies the compression filter to be used for images. Ignored if <i>@AutoFilterImages</i> = "true" or if <i>@EncodeImages</i> = "false".</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>CCITTFaxEncode</i> – Used to select CCITT Group 3 or 4 facsimile encoding. Used only if <i>@ImageType</i> = "Monochrome". <i>DCTEncode</i> – Used to select JPEG compression. <i>FlateEncode</i> – Used to select ZIP compression. <i>JBIG2Encode</i> – Used to select JBIG2 encoding. Used only if <i>@ImageType</i> = "Monochrome". <i>JPEG2000</i> – Used to select JPEG2000/Wavelet compression. <i>LZWEncode</i> – LZW Compression. <i>PackBits</i> – A simple byte-oriented run length scheme.
<i>ImageResolution</i> ?	double	Specifies the minimum resolution for downsampled color images in dots per inch. This value is used only when <i>@DownsampleImages</i> = "true". The application downsamples only images that are above that resolution to that actual resolution.
<i>ImageType</i>	enumerations	<p>Specifies the kind of images that are to be manipulated.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>All</i> – image compression is applied to all image types. <i>Color</i> <i>Grayscale</i> <i>Monochrome</i>
<i>JPXQuality</i> ?	integer	Specifies the image quality. Valid values are greater than or equal to one (1) and less than or equal to 100. One (1) means lowest quality (highest compression), 99 means visually lossless compression, and 100 means numerically lossless compression.
<i>CCITTFaxParams</i> ?	element	The equivalent of the PostScript <i>Rows</i> and <i>BlackIs1</i> parameters, which are implicit in the raster data to be compressed.
<i>DCTParams</i> ?	element	Provides the equivalents of the PostScript <i>Columns</i> , <i>Rows</i> and <i>Colors</i> Attributes, which are assumed to be implicit in the raster data to be compressed.
<i>FlateParams</i> ?	element	The equivalent of the PostScript <i>Columns</i> , <i>BitsPerComponent</i> and <i>Colors</i> parameters, which are implicit in the raster data to be compressed.
<i>JBIG2Params</i> ?	element	Provides the JBIG2 compression parameters.
<i>JPEG2000Params</i> ?	element	Provides the JPEG2000 compression parameters.
<i>LZWParams</i> ?	element	The equivalent of the PostScript <i>Columns</i> , <i>BitsPerComponent</i> and <i>Colors</i> parameters, which are implicit in the raster data to be compressed

8.49.2 Element: CCITTFaxParams

Table 8-92: CCITTFaxParams Element

Name	Data Type	Description
<i>EncodedByteAlign ?</i>	boolean	A flag indicating whether the CCITTFaxEncode filter inserts an extra 0 bits before each encoded line so that the line begins on a byte boundary.
<i>EndOfBlock ?</i>	boolean	A flag indicating whether the CCITTFaxEncode filter appends an end-of-block pattern to the encoded data
<i>EndOfLine ?</i>	boolean	A flag indicating whether the CCITTFaxEncode filter prefixes an end-of-line bit pattern to each line of encoded data.
<i>K ?</i>	integer	An integer that selects the encoding scheme to be used. < 0 – Pure two-dimensional encoding (Group 4, TIFF Compression = 4) = 0 – Pure one-dimensional encoding (Group 3, 1-D, TIFF Compression = 2) > 0 – Mixed one- and two-dimensional encoding (Group 3, 2-D, TIFF Compression = 3), in which a line encoded one-dimensionally can be followed by at most K – 1 lines encoded two-dimensionally
<i>Uncompressed ?</i>	boolean	A flag to indicate whether the file generated can use uncompressed encoding when advantageous.

8.49.3 Element: DCTParams

Table 8-93: DCTParams Element (Sheet 1 of 2)

Name	Data Type	Description
<i>ColorTransform ?</i>	enumeration	Color transformation algorithm. Allowed values are: <i>None</i> – Colors are not to be transformed. <i>YUV</i> – RGB raster values are to be transformed to YUV before encoding and from YUV to RGB after decoding. If four channels are present, transform CMYK values to YUVK before encoding and from YUVK to CMYK after decoding. <i>Automatic</i> – "YUV" for 3-channel raster data, "None" otherwise. Note: YUV is equivalent to YCbCr in TIFF terminology.
<i>HSamples ?</i>	IntegerList	A sequence of horizontal sampling factors—one entry per color channel in the raster data. If not specified, the implied default is "1" for every channel.
<i>HuffTable ?</i>	DoubleList	Huffman tables for DC and AC components. If present, there SHALL be at least one HuffTable element for each color channel.
<i>QFactor ?</i>	double	A scale factor applied to the elements of <i>@QuantTable</i> .
<i>QuantTable ?</i>	DoubleList	Quantization tables. If present, there SHALL be one <i>@QuantTable</i> entry for each color channel.

Table 8-93: DCTParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>SourceCSs</i>	enumerations	<p>Identifies which of the incoming color spaces will be operated on.</p> <p>Allowed values are from: Table 8-129, “SourceCSs Attribute Values” on page 696.</p> <p>Note: JDF 1.1 defined that CalRGB be treated as RGB, CalGray as Gray, and ICC-Based color spaces as one of Gray, RGB or CMYK depending on the number of channels.</p> <p>Note: In JDF 1.2, the data type was erroneously specified as enumeration, not enumerations.</p>
<i>VSamples ?</i>	IntegerList	A sequence of vertical sampling factors—one entry per color channel in the raster data. If not specified, the implied default is "1" for every channel.

When the **DCTParams** Element is a Subelement of **ImageCompressionParams** used in a **Rendering** Process to generate TIFF files, YUV is equivalent to YCbCr in TIFF terminology. The HSamples and VSamples values are used to set YCbCrSubSampling or CIELabSubSampling. This means that they are only relevant for data supplied as Lab, or data where *@ColorTransform* is "YUV"; that the first element SHALL be 1 in each case; that the fourth element SHALL be 1 where CMYK data is to be compressed; and that the second and third elements SHALL equal each other.

— Attribute: **SourceCSs**

Table 8-94: SourceCSs Attribute Values

Value	Description
<i>Calibrated</i>	Operates on CalGray and CalRGB color spaces.
<i>CIEBased</i>	Operates on CIE-Based color spaces (CIEBasedA, CIEbasedABC, CIEBasedDEF, CIE-BasedDEFG).
<i>CMYK</i>	Operates on characterized and uncharacterized DeviceCMYK.
<i>DeviceN</i>	Identifies the source color encoding as a DeviceN color space. The specific DeviceN color space to operate on is defined in the DeviceNSpace Resource. If this value is specified then <i>DeviceNSpace</i> SHALL also be present.
<i>DevIndep</i>	Operates on Device independent color spaces (equivalent to Calibrated or CIE-Based or ICC-Based or Lab or YUV).
<i>Gray</i>	Operates on characterized and uncharacterized DeviceGray.
<i>ICCBased</i>	Operates on color spaces defined using ICC profiles. ICC-Based includes EPS, TIFF or PICT files with embedded ICC profiles. See [ICC.1].
<i>Lab</i>	Operates on Lab.
<i>RGB</i>	Operates on characterized and uncharacterized DeviceRGB
<i>Separation</i>	Operates on Separation color spaces (spot colors). The specific separation(s) to operate on are defined in the <i>@SeparationSpecSeparations</i> Attribute. If no <i>@Separations</i> Attribute is defined, the operation will operate on all the separation color spaces in the input RunList .
<i>YUV</i>	Operates on YUV (Also known as YCbCr). See [CCIR601-2].

8.49.4 Element: FlateParams

Table 8-95: FlateParams Element

Name	Data Type	Description
<i>Effort</i> ?	integer	A code controlling the amount of memory used and the execution speed for Flate compression. Allowed values range from 0 to 9. A value of 0 compresses rapidly but not tightly, using little auxiliary memory. A value of 9 compresses slowly but as tightly as possible, using a large amount of auxiliary memory.
<i>Predictor</i> ?	integer	<p>A code that selects the predictor function:</p> <p>Note: On 1X PNG predictors, these values select the specific PNG predictor function(s) to be used, as indicated above. When decoding the predictor function is explicitly encoded in the incoming data.</p> <p>Values include:</p> <ul style="list-style-type: none"> 1 – No predictor (normal encoding or decoding). 2 – TIFF Predictor 2. 10 – PNG predictor, None function. 11 – PNG predictor, Sub function. 12 – PNG predictor, Up function. 13 – PNG predictor, Average function. 14 – PNG predictor, Path function. 15 – PNG predictor in which the encoding filter automatically chooses the optimum function separately for each row.

8.49.5 Element: JBIG2Params

Table 8-96: JBIG2Params Element

Name	Data Type	Description
<i>JBIG2Lossless</i> ?	boolean	If "true" requires JBIG2 compressed images to retain the exact representation of the original image without loss.

8.49.6 Element: JPEG2000Params

Table 8-97: JPEG2000Params Element (Sheet 1 of 2)

Name	Data Type	Description
<i>CodeBlockSize</i> ?	integer	The nominal code block width and height. SHALL be a power of 2.
<i>LayerRates</i> ?	DoubleList	<p>Compression bit ratio for each layer. If specified, there SHALL be the same number of doubles in this list as <i>@LayersPerTile</i> in ascending order.</p> <p>Small values correspond to maximum compression and 1.0 corresponds to no compression (lossless).</p> <p>If available, <i>@LayerRates</i> SHOULD be supplied.</p>
<i>LayersPerTile</i> ?	integer	Specifies the number of quality layers per tile at the same resolution.
<i>NumResolutions</i> ?	integer	The number of resolution levels encoded in the file.

Table 8-97: JPEG2000Params Element (Sheet 2 of 2)

Name	Data Type	Description
<i>ProgressionOrder</i> ?	enumeration	<p>Per tile progression order.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>LRC</i>P – layer-resolution-component-position progressive (i.e., rate scalable). <i>RLCP</i> – Resolution-layer-component-position progressive (i.e., resolution scalable). <i>RPCL</i> – Resolution-position-component-layer progressive. <i>PCRL</i> – Position-component-resolution-layer progressive. <i>CPRL</i> – Component-position-resolution-layer progressive.
<i>TileSize</i> ?	XYPair	The width and height of each encoding tile. If not specified the image is considered to be a single tile.

8.49.7 Element: LZWParams

Table 8-98: LZWParams Element

Name	Data Type	Description
<i>EarlyChange</i> ?	integer	<p>A code indicating when to increase the code word length. The TIFF specification can be interpreted to imply that code word length increases are postponed as long as possible. However, some existing implementations of LZW increase the code word length one code word earlier than necessary. The PostScript language supports both interpretations. If @<i>EarlyChange</i> is "0", code word length increases are postponed as long as possible. If it is "1", they occur one code word early.</p> <p>Note: The default SHOULD NOT be used when this LZWParams Element is in ImageCompressionParams used as an Input Resource to a FormatConversion Process that is creating TIFF files.</p>
<i>Predictor</i> ?	integer	<p>A code that selects the predictor function:</p> <ul style="list-style-type: none"> 1 – No predictor (normal encoding or decoding). 2 – TIFF Predictor 2. 10 – PNG predictor, None function. 11 – PNG predictor, Sub function. 12 – PNG predictor, Up function. 13 – PNG predictor, Average function. 14 – PNG predictor, Path function. 15 – PNG predictor in which the encoding filter automatically chooses the optimum function separately for each row. <p>Note: On 1X PNG predictors, these values select the specific PNG predictor function(s) to be used, as indicated above. When decoding, the predictor function is explicitly encoded in the incoming data.</p>

8.50 ImageEnhancementParams

Resource Properties

Resource referenced by: —

—Input of Processes:

Output of Processes: —

Table 8-99: ImageEnhancementParams Resource

Name	Data Type	Description
ImageEnhancementOp *	element	Individual enhancement operations. The XML order of ImageEnhancementOp Elements is significant. When multiple elements apply to the same object, they are applied in that XML order.

8.50.1 Element: ImageEnhancementOp

Table 8-100: ImageEnhancementOp Element

Name	Data Type	Description
<i>ObjectTags</i> ?	NMTOKENS	<p>Tags associated with individual objects that this ImageEnhancementOp SHALL be applied to. Each tag specified in <i>@ObjectTags</i> is logically added with the object type(s) specified by <i>@SourceObjects</i>, enabling first qualification by object type (such as image), and then tags associated with those objects.</p> <p>The values of <i>@ObjectTags</i> depends on the PDL that the color correction is applied to.</p> <p><i>@ObjectTags</i> SHALL apply only to objects whose tag pool includes all the tags in the value of <i>@ObjectTags</i>.</p>
<i>Operation</i>	NMTOKEN	<p>Individual enhancement operation name.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Sharpening</i> – Image Sharpening <i>Blurring</i> – Image Blurring <i>RedEyeRemoval</i> <i>BestGuess</i> – Best guess automated improvements based on image analysis
<i>OperationDetails</i> ?	string	Additional details of the <i>@Operation</i> . The values are implementation specific.
<i>SourceObjects</i> ?	enumerations	<p>Identifies which class(es) of incoming graphical objects will be operated on.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>All</i> <i>ImagePhotographic</i> – Contone images. <i>ImageScreenShot</i> – Images largely comprised of rasterized vector art. <i>LineArt</i> – Vector objects other than text. <i>SmoothShades</i> – Gradients and blends. <i>Text</i>

8.51 ImageSetterParams

This Resource specifies the settings for the imagesetter. A number of settings are OEM-specific, while others are so widely used they MAY be supported between vendors. Both filmsetter settings and platesetter settings are described with this Resource.

Resource Properties

Resource referenced by: **PreviewGenerationParams**

Intent Pairing: **ContentCheckIntent**

Input of Processes: **ImageSetting**

Output of Processes: —

Table 8-101: ImageSetterParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>AdvanceDistance</i> ?	double	Additional media advancement beyond the media dimensions on a Web-Fed Device.
<i>BurnOutArea</i> ?	XYPair	Size of the burnout area. The area defined by <i>@BurnOutArea</i> is exposed, regardless of the size of the image. If not specified or "0 0", only the area defined by the image is exposed.
<i>CenterAcross</i> ?	enumeration	<p>Specifies the axis around which a Device is to center an image if the Device is capable of doing so.</p> <p>Allowed values are:</p> <p><i>None</i> – Do not center.</p> <p><i>FeedDirection</i> – Image is centered around the feed-direction axis.</p> <p><i>MediaWidth</i> – Image is centered around the media-width axis.</p> <p><i>Both</i> – Image is centered around both axes.</p>
<i>CutMedia</i> ?	boolean	Indicates whether or not to cut the media (Web-Fed).
<i>ManualFeed</i> ?	boolean	Indicates whether the media will be fed manually.
<i>MirrorAround</i> ?	enumeration	<p>This Attribute specifies the axis around which a Device SHALL mirror an image if the Device is capable of doing so.</p> <p>Allowed values are:</p> <p><i>None</i> – Do not mirror the image.</p> <p><i>FeedDirection</i> – Image is mirrored around the feed-direction axis.</p> <p><i>MediaWidth</i> – Image is mirrored around the media-width axis.</p> <p><i>Both</i> – Image is mirrored around both possible axes.</p>
<i>Polarity</i> ?	enumeration	<p>Some Devices can invert the image (in hardware).</p> <p>Allowed values are:</p> <p><i>Positive</i></p> <p><i>Negative</i></p>
<i>Resolution</i> ?	XYPair	Resolution of the output. If not specified, the default is taken from the resolution of the input ByteMap.
<i>RollCut</i> ?	double	Length of media to be cut off of a Roll, in points.

Table 8-101: ImageSetterParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>TransferCurve</i> ?	Transfer-Function	Area coverage correction of the Device.
<i>FitPolicy</i> ?	element	Describes the hardware image fitting algorithms. Allows printing even if the size of the imageable area of the media does not match the requirements of the data.

8.52 Ink

Resource describing what kind of ink or other colorant (e.g., toner, varnish) is to be used during printing or varnishing. The default unit of measurement for **Ink** is *@Unit* = “g” (gram).

Resource Properties

Resource referenced by: —

Input of Processes: *ConventionalPrinting, DigitalPrinting, Varnishing*

Output of Processes: —

8.53 InkZoneCalculationParams

Table 8-102: Ink Resource

Name	Data Type	Description
<i>InkName</i> ?	string	The fully qualified ink name including the ink <i>@Family</i> name. For instance, "PANTONE 138 C" is a member of the PANTONE family.
<i>InkType</i> ?	NMTOKENS	<p><i>@InkType</i> SHALL list specific ink type and qualities. e.g. <i>@InkType</i>="Gloss Relief Varnish"</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Aqueous</i> – attribute of the ink <i>Dull</i> – attribute of the ink <i>Gloss</i> – attribute of the ink <i>HKS</i> – ink <i>ISO</i> – ink [ISO2846-1:1997] (used by SWOP) <i>InkJet</i> – ink <i>Latex</i> – liquid that is similar to ink <i>Metallic</i> – attribute of the ink <i>PANTONE</i> – ink <i>Protective</i> – attribute of the ink <i>Relief</i> – attribute of the ink <i>Silicon</i> – liquid that is similar to ink <i>Satin</i> – attribute of the ink <i>Toyo</i> – ink <i>Toner</i> – liquid that is similar to ink <i>UV</i> – attribute of the ink <i>Varnish</i> – liquid that is similar to ink
<i>SpecificYield</i> ?	double	Weight per area at total coverage in g/m ² .

This Resource specifies the parameters for the *InkZoneCalculation* Process.

Resource Properties

Resource referenced by: —

Input of Processes: *InkZoneCalculation*

Output of Processes: —

Table 8-103: InkZoneCalculationParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>PrintableArea</i> ?	rectangle	<p>Position and size of the printable area of the print cylinder in the coordinates of the <i>Preview</i> Resource.</p> <p>The Partition <i>@TileID</i> SHALL be used for each plate together with this Attribute in case of multiple plates per cylinder. Multiple plates per cylinder MAY be used in Web Printing.</p> <p>The default case is to specify a rectangle that encompasses the complete image to be printed.</p>
<i>ZoneHeight</i> ?	double	The width of one zone in the feed direction of the printing Machine being used.

Table 8-103: InkZoneCalculationParams Resource (Sheet 2 of 2)

Name	Data Type	Description
Zones ?	integer	The number of ink zones of the press.
ZonesY ?	integer	Number of ink zones in feed direction of the press.
ZoneWidth ?	double	The width of one zone of the printing Machine being used. Typically, the width of a zone is the width of an ink slide.
Device ?	element	Device provides a reference to the press that the InkZoneProfile is defined for and is used to gather information about ink zone geometry.

8.54 InkZoneProfile

This Resource specifies ink zone settings that are specific to the geometry of the printing Device being used. **InkZoneProfile** Elements are independent of the Device details.

Resource Properties

Resource referenced by:

—

Input of Processes:

ConventionalPrinting

Output of Processes:

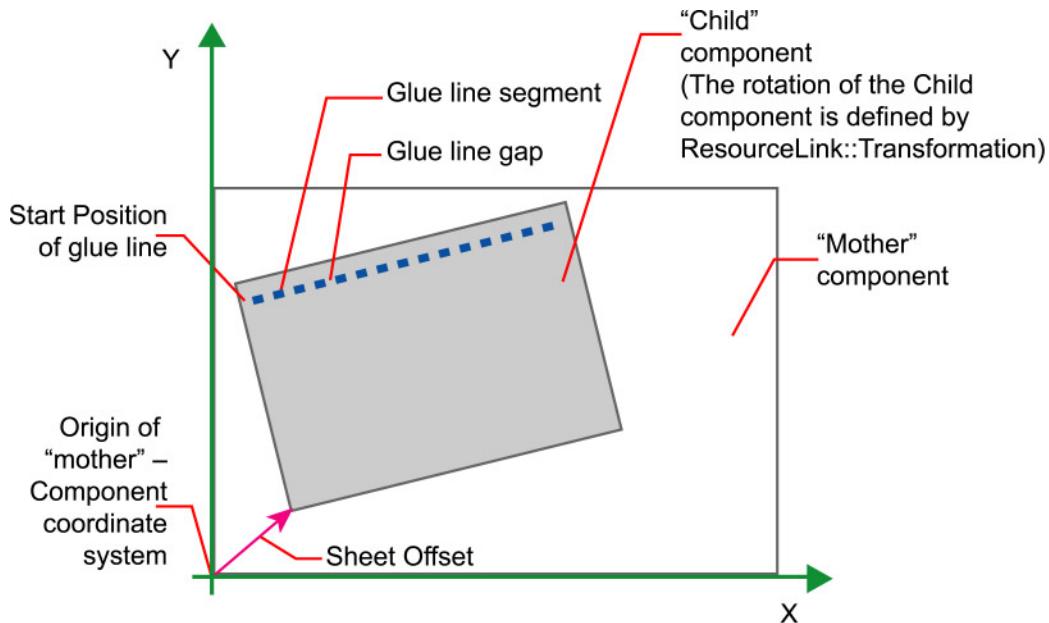
InkZoneCalculation

Table 8-104: InkZoneProfile Resource

Name	Data Type	Description
ZoneHeight ?	double	The width of one zone in the feed direction of the printing Machine being used.
ZoneSettingsX	DoubleList	Each entry of the @ZoneSettingsX Attribute is the value of one ink zone. The first entry is the first zone, and the number of entries equals the number of zones of the printing Device being used. Allowed values are in the range [0..1] where 0 is no ink and 1 is 100% coverage.
ZoneSettingsY ?	DoubleList	Each entry of the @ZoneSettingsY Attribute is the value of one ink zone in Y Direction. The first entry is the first zone, and the number of entries equals the number of zones of the printing Device being used. Allowed values are in the range [0..1] where 0 is no ink and 1 is 100% coverage.
ZoneWidth	double	The width of one zone of the printing Machine being used. Typically, the width of a zone is the width of an ink slide.

8.55 InsertingParams

This Resource specifies the parameters for the **Inserting** Process. Figure 7.13 shows the various components involved in an inserting Process, and how they interact.

Figure 8-31: Parameters and coordinate system used for Inserting

The process coordinate system is defined as follows: The Y-axis is aligned with the binding edge and increases from the registered edge to the edge opposite the registered edge. The X-axis, meanwhile, is aligned with the registered edge. It increases from the binding edge to the edge opposite the binding edge, which is the product front edge.

Resource Properties

Resource referenced by:

InsertingIntent

Intent Pairing:

Inserting

Input of Processes:

—

Output of Processes:

—

Table 8-105: InsertingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>FinishedPage</i> ?	integer	Finished Page number of the mother Component on which the child Component has to be placed. <i>@FinishedPage</i> SHALL NOT be specified unless <i>@InsertLocation</i> = " <i>FinishedPage</i> ". Corresponds to <i>@Folio</i> on InsertingIntent .
<i>InsertLocation</i>	enumeration	<p>Where to place the "child" Sheet.</p> <p>Allowed values are:</p> <p><i>Back</i> <i>FinishedPage</i> – Place the child exactly onto the page specified in <i>@FinishedPage</i>.</p> <p><i>Front</i> <i>Overfold</i> – Place onto the overfold. Replaces "<i>OverfoldLeft</i>" and "<i>OverfoldRight</i>".</p>
<i>Method</i> ?	enumeration	<p>Inserting method.</p> <p>Allowed values are:</p> <p><i>BindIn</i> – Apply glue to fasten the insert.</p> <p><i>BlowIn</i> – Loose insert.</p>

Table 8-105: InsertingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
GlueLine *	element	Array of all GlueLine Elements. The coordinate system is defined by the mother Component .

Location of Inserts

The following graphics depict the various values of **InsertingParams/@InsertLocation**:

Table 8-106: Location of Inserts

Front	Back	Overfold	Finished Page

Child on *Front* of mother component — is used for fixed inserts (e.g., gluing of inserts and so forth on Signatures).

Child on *Back* of mother component — is used for fixed inserts (e.g., gluing of inserts on Signatures).

The mother component is opened at the overfold and the child is placed in the center of the of the mother. *Overfold* is used for loose inserts (e.g., inserts into newspapers).

Child on "FinishedPage" X of mother component — can be used for loose and fixed inserts.

8.56 InterpretingParams

The **InterpretingParams** Resource contains the parameters needed to interpret PDL pages. The Resource itself is a generic Resource that contains Attributes that are relevant to all PDLs. PDL-specific instances of **InterpretingParams** Resources MAY be included as Subelements of this generic Resource. This specification defines one additional PDL-specific Resource instance: **PDFInterpretingParams**.

Resource Properties

Resource referenced by: —

Intent Pairing: **ContentCheckIntent**

Input of Processes: **Interpreting**

Output of Processes: —

Table 8-107: InterpretingParams Resource (Sheet 1 of 3)

Name	Data Type	Description
<i>Center</i> ?	boolean	Indicates whether or not the finished page image is to be centered within the imageable area of the media. The <i>@Center</i> is ignored if <i>FitPolicy/@SizePolicy</i> = "ClipToMaxPage" and clipping is specified.

Table 8-107: InterpretingParams Resource (Sheet 2 of 3)

Name	Data Type	Description
<i>FilmRef?</i>	IDREF	Reference to film Media . This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during Interpreting .
<i>MirrorAround?</i>	enumeration	This Attribute specifies the axis around which a RIP is to mirror an image. Note that this is mirroring in the RIP and not in the hardware of the output Device. Allowed values are: <i>None</i> – The default. <i>FeedDirection</i> – Image is mirrored around the feed-direction axis. <i>MediaWidth</i> – Image is mirrored around the media-width axis. <i>Both</i> – Image is mirrored around both possible axes.
<i>PaperRef?</i>	IDREF	Reference to final paper Media . This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during Interpreting .
<i>PlateRef?</i>	IDREF	Reference to plate Media . This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during Interpreting .
<i>Polarity?</i>	enumeration	The image SHALL be RIPed in the specified polarity. Note that this is a polarity change in the RIP and not a polarity change in the hardware of the output Device. Allowed values are: <i>Positive</i> <i>Negative</i>
<i>PrintQuality?</i>	enumeration	Generic switch for setting the quality of an otherwise inaccessible Device. Allowed values are: <i>High</i> – Highest quality available on the printer. <i>Normal</i> – The default quality provided by the printer. <i>Draft</i> – Lowest quality available on the printer.
<i>ProofPaperRef?</i>	IDREF	Reference to paper Media used for proofing. This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during Interpreting .
<i>Scaling?</i>	XYPair	A pair of positive real values that indicates the scaling factor for the page contents. Values between 0 and 1 specify that the contents are to be reduced, while values greater than 1 specify that the contents are to be expanded. This Attribute is ignored if <i>@FitToPage = "true"</i> . Any scaling defined in FitPolicy SHALL be applied after the scaling defined by this Attribute.

Table 8-107: InterpretingParams Resource (Sheet 3 of 3)

Name	Data Type	Description
<i>ScalingOrigin</i> ?	XYPair	A pair of real values that identifies the point in the unscaled PDL page that remains at the same position after scaling. This point is defined in the coordinate system of the PDL page. For example, The <i>@ScalingOrigin</i> of a PDL page with dimensions "300 400" scaled from the PDL page center would be "150 200", regardless of the value of <i>@Scaling</i> .
FitPolicy ?	element	Allows printing even if the size of the imageable area of the media does not match the requirements of the data. This replaces the deprecated <i>@FitToPage</i> Attribute.
InterpretingDetails ?	element	Container for interpreter-specific details.
ObjectResolution *	element	Indicates the resolution at which the PDL contents will be interpreted in DPI. These Elements MAY be different from the ObjectResolution Elements provided in the Resource.
PDFInterpretingParams ?	element	Details of interpreting for PDF.
PDLResourceAlias ?	element	

8.56.1 Element: InterpretingDetails

InterpretingDetails contains PDL-specific instructions for an interpreter.

Table 8-108: InterpretingDetails Element

Name	Data Type	Description
<i>MinLineWidth</i> ?	double	If present, this attribute specifies the minimum width in points for PDL line objects. If a line is defined with a width smaller than this value it SHALL be adjusted to a line width equal to this value. Note: This attribute is useful for managing the consistency of thin lines across different digital printing systems that have varying imaging resolutions.

8.56.2 Element: PDFInterpretingParams

Table 8-109: PDFInterpretingParams Element (Sheet 1 of 2)

Name	Data Type	Description
<i>EmitPDFBG</i> ?	boolean	Indicates whether BlackGeneration functions are to be emitted.
<i>EmitPDFHalftones</i> ?	boolean	Indicates whether Halftones are to be emitted.
<i>EmitPDFTransfers</i> ?	boolean	Indicates whether Transfer functions are to be emitted
<i>EmitPDFUCR</i> ?	boolean	Indicates whether UnderColorRemoval functions are to be emitted.
<i>HonorPDFOverprint</i> ?	boolean	Indicates whether or not overprint settings in the file will be honored. If "true", the setting for overprint will be honored. If "false", it is expected that the Device does not directly support overprint and that the PDF is preprocessed to simulate the effect of the overprint settings

Table 8-109: PDFInterpretingParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>ICCCColorAsDeviceColor</i> ?	boolean	Indicates whether colors specified by ICC color spaces are to be treated as Device colorants.
<i>OCGIntent</i> ?	NMTOKEN	If $\text{@OCGDefault} = \text{"FromPDF"}$, then the value of @OCGIntent sets the intent for which OCGs are to be selected. Values include: <i>Design</i> – as described in [PDF1.6]. <i>View</i> – as described in [PDF1.6].
<i>OCGProcess</i> ?	enumeration	If $\text{@OCGDefault} = \text{"FromPDF"}$, then the value of @OCGProcess sets the purpose for which the Interpreting Process is being performed. This, in turn, sets which value from a relevant optional content usage dictionary is to be used to determine whether each OCG is included in the RunList . Allowed values are: <i>Export</i> – PDF ExportState in the Export subdictionary. <i>Print</i> – PDF PrintState in the Print subdictionary <i>View</i> – PDF ViewState in the View subdictionary.
<i>OCGZoom</i> ?	double	If $\text{@OCGDefault} = \text{"FromPDF"}$, then the value of @OCGZoom sets the magnification to be assumed in comparisons with the Zoom dictionary in a relevant optional content usage dictionary to determine whether each OCG is included in the RunList . A @OCGZoom value of 1.0 is assumed to be a magnification of 100%.
<i>PrintPDFAnnotations</i> ?	boolean	Indicates whether the contents of annotations on PDF pages SHALL be included in the output. This only refers to annotations that are set to print in the PDF file excluding trap annotations. Trap annotations are controlled with $\text{@PrintTrapAnnotations}$.
<i>PrintTrapAnnotations</i> ?	boolean	Indicates whether the contents of trap annotations on PDF pages SHALL be included in the output.
<i>TransparencyRenderingQuality</i> ?	double	Values are 0 to 1. A value of 0 represents the lowest allowable quality; 1 represents the highest desired quality.
<i>OCGControl</i> *	element	<i>OCGControl</i> provides a list of the OCGs (layers) that are to be explicitly included or excluded in the RunList . Any OCGs not listed in an <i>OCGControl</i> Element SHALL follow the rules set by the referenced PDF.
<i>ReferenceXObjParams</i> ?	element	Describes how the interpreter should handle PDF Reference XObjects

8.56.3 Element: ReferenceXObjParams

Table 8-110: ReferenceXObjParams Element

Name	Data Type	Description
<i>Mode</i>	NMTOKEN	<p>Specifies how to handle a Reference XObject's reference.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Ignore</i> – the reference is ignored, and no content is imaged for that Reference XObject. If proxy content is supplied with the Reference XObject, it is imaged. <i>ResolveAlways</i> – an attempt is made to resolve the reference, and image the graphics described by that reference. <i>ResolveIfPDFX5</i> – an attempt is made to resolve the reference ONLY if the PDF file is a valid PDF/X-5 file, AND the referenced file passes the criteria stated in section 8.4 of ISO 15930-8 (PDF/X-5).
<i>FileSpec (SearchPath) *</i>	element	An ordered list of <i>FileSpec</i> elements that specify search paths to search when an XObject provides a relative file specification for its target file. If not specified, then the directory that contains the PDF file being interpreted SHALL be searched, and SHALL NOT be searched recursively.

8.56.4 Element: PDLResourceAlias

This Parameter provides a mechanism for referencing Resources that occur in files, or that are expected to be provided by Devices. Prepress and printing Processes have traditionally used the word “Resource” to refer to reusable data structures that are needed to perform Processes. Examples of such Resources include fonts, halftones and functions. The formats of these Resources are defined within PDLs, and instances of these Resources can occur within PDL files or can be provided by Devices.

XJDF does not provide a syntax for defining such Resources directly within a Job. Instead, Resources continue to occur within PDL files and continue to be provided by Devices. However, since it is necessary to be able to refer to these Resources from XJDF Jobs, the *PDLResourceAlias* element is provided to fulfill this need.

Table 8-111: PDLResourceAlias Element

Name	Data Type	Description
<i>FileSpec?</i>	element	Location of the file containing the PDL Resource. If <i>FileSpec</i> is absent, the Device is expected to provide the Resource defined by this <i>PDLResourceAlias</i> element.
<i>ResourceType</i>	string	The type of PDL Resource that is referenced. The semantic of this Attribute is defined by the PDL.
<i>SourceName?</i>	string	The name of the Resource in the file referenced by the <i>FileSpec</i> or by the Device.

8.56.5 More about PDFInterpretingParams

8.56.5.1 PDF Optional Content Groups

The order of OCGControl Elements has no effect; the Z-order of graphic elements that make up each optional content group (the term layer is misleading in this regard) within the PDF file defines the drawing order of those graphic elements.

Any preferences recorded in OCGs within the PDF file as to whether that OCG are to be displayed or not will be ignored if that OCG is referenced from an **OCGControl Element**, or if **@OCGDefault** is either "*Include*" or "*Exclude*"; PDF preferences are only applied when **@OCGDefault = "FromPDF"**.

If **@OCGDefault = "FromPDF"**, the state of all OCGs explicitly referenced from **OCGControl Elements** SHALL be set before determining the state of any remaining OCGs.

All controls for OCGs in **XJDF** address OCGs directly, and not optional Content Member Dictionaries (OCMDs do not have unique names).

NOTE: [PDF1.6] does not state that all OCGs SHALL have unique names. It is therefore possible for a single PDF file to contain multiple OCGs with the same name. When **OCGControl/@OCGName** refers to multiple OCGs in a file, they will all be explicitly included or excluded together.

8.57 JacketingParams

Description of the setup of the jacketing machinery. Jacket height and width (1 and 4 in the Figure 8-39) are specified within the **Component** that describes the jacket.

Figure 8-32: Setup of the Jacketing Machinery

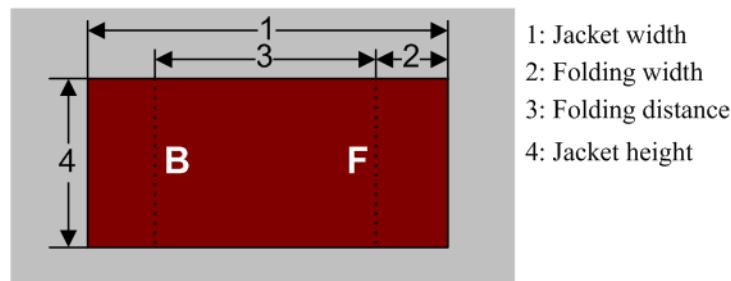
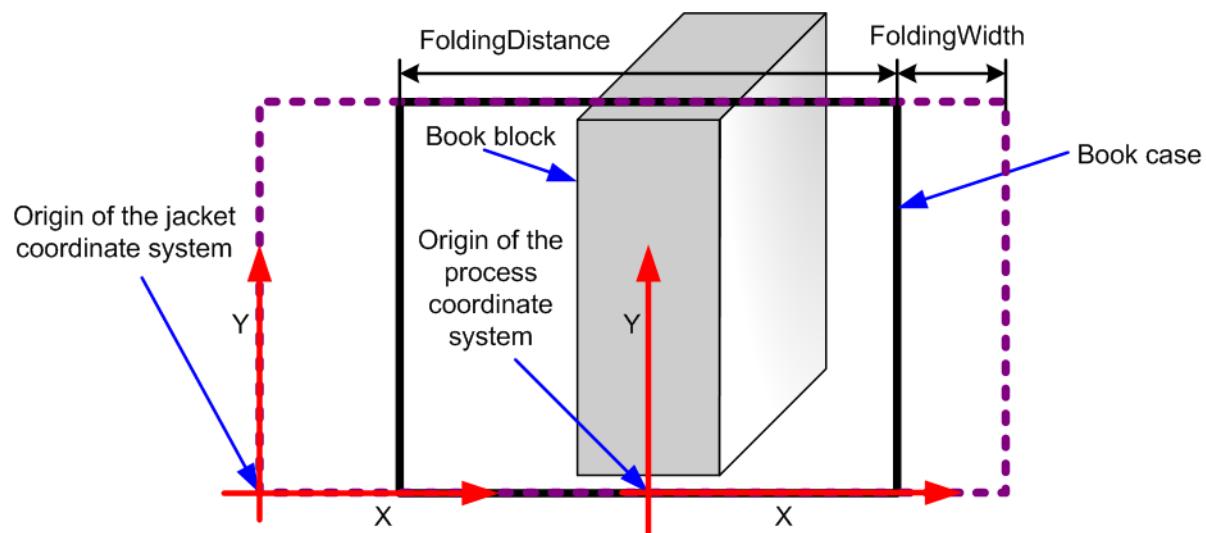


Figure 8-33: Parameters and coordinate system for jacketing



Resource Properties

Resource referenced by:

Intent Pairing:

Input of Processes:

—

BindingIntent
Jacketing

Output of Processes: —**Table 8-112: JacketingParams Resource**

Name	Data Type	Description
<i>FoldingDistance</i> ?	double	Distance from the fold at <i>@FoldingWidth</i> to the other fold. If not specified, it defaults to width of the Jacket minus two times <i>@FoldingWidth</i> (symmetrical folds).
<i>FoldingWidth</i>	double	Definition of the dimension of the folding width of the front cover fold (see <i>@FoldingWidth</i> in the picture above). All other measurements are implied by the dimensions of the book.

8.58 LabelingParams

LabelingParams defines the details of the **Labeling** Process.

Resource Properties

Resource referenced by: —

Input of Processes: *Labeling*

Output of Processes: —

Table 8-113: LabelingParams Resource

Name	Data Type	Description
<i>Application</i> ?	NMTOKEN	Application method of the label. Values include: <i>Glue</i> – Glued onto the component. <i>Loose</i> – Loosely laid onto the component. <i>SelfAdhesive</i> – Self adhesive label. <i>Staple</i> – Stapled onto the component.
<i>CTM</i> ?	matrix	Position and orientation of the label lower-left-corner relative to the lower left corner of the component surface as defined by <i>@Position</i> .
<i>Position</i> ?	enumeration	Position of the label on the bundle. Allowed values are: <i>Back</i> <i>Bottom</i> <i>Front</i> <i>Left</i> <i>Right</i> <i>Top</i>
FileSpec (<i>AddressList</i>) ?	element	A FileSpec Resource pointing to an address list. The format of the referenced mailing list is implementation dependent.

8.59 LaminatingParams

This Resource specifies the parameters needed for laminating.

Resource Properties

Resource referenced by: —

Intent Pairing: **LaminatingIntent**
Input of Processes: **Laminating**
Output of Processes: —

Table 8-114: LaminatingParams Resource

Name	Data Type	Description
<i>AdhesiveType</i> ?	string	Type of adhesive used. Valid only when @ <i>LaminatingMethod</i> = "DispersionGlue".
<i>GapList</i> ?	DoubleList	List of non-laminated gap positions in the X direction of the laminating tool in the coordinate system of the Component . The zero-based even entries define the absolute position of the start of a gap, and the odd entries define the end of a gap. If not specified, the complete area defined by @ <i>LaminatingBox</i> is laminated.
<i>HardenerType</i> ?	string	Type of hardener used. Valid only when @ <i>LaminatingMethod</i> = "DispersionGlue".
<i>LaminatingBox</i> ?	rectangle	Area on the Component to be laminated.
<i>LaminatingMethod</i> ?	enumeration	Laminating technology that is applied. Allowed values are: <i>CompoundFoil</i> <i>DispersionGlue</i> <i>Fusing</i>
<i>ModuleID</i> ?	integer NMOKEN	Identifier of the laminating Module in the Press. See ConventionalPrintingParams .
<i>NipWidth</i> ?	double	Width of the nip in points to be formed between the fusing rollers and the component in the Laminating Process.
<i>Temperature</i> ?	double	Temperature used in the Laminating Process, in ° Centigrade.

8.60 Layout

Represents the root of the layout structure. The **Layout** is used both for fixed-layout and for automated printing.

Resource Properties

Resource referenced by: **Component**
Input of Processes: **ConventionalPrinting**, **DigitalPrinting**, **Imposition**,
InkZoneCalculation
Output of Processes: **Stripping**

Table 8-115: Layout Resource (Sheet 1 of 5)

Name	Data Type	Description
<i>Automated</i> ?	boolean	If "true", the Imposition Process is expected to perform automated imposition. Layout/@Automated SHALL only be specified in the root Partition of Layout . Default value is: "false" in the root Partition of Layout

Table 8-115: Layout Resource (Sheet 2 of 5)

Name	Data Type	Description
<i>BaseOrdReset</i> ?	enumeration	<p>Policy about how the <i>@Ord</i> Attribute of an entry SHALL be calculated when extracting a page from a RunList and positioning it in the Layout.</p> <p>Allowed values are:</p> <p><i>PagePool</i> – The Base Ord is reset to point to the first page entry of the Page Pool at the beginning of each Page Pool processed by the Imposition Template.</p> <p><i>PagePoolList</i> – At the beginning of processing of the Imposition Template, the Base Ord is reset to point to the first page entry of the first Page Pool to be processed by the Imposition Template. This results in all Page Pools that will be processed by the Imposition Template to be treated as a Page Pool List.</p>
<i>FilmRef</i> ?	IDREF	Reference to film Media . This Resource provides a description of the physical media which will be marked.
<i>LockOrigins</i> ?	boolean	Determines the relationship of the coordinate systems for front and back surfaces. When " <i>false</i> ", all contents for all surfaces are transformed into the first quadrant, in which the origin is at the lower left corner of the surface. When " <i>true</i> ", contents for the front surface are imaged into the first quadrant (as above), but contents for the back surface are imaged into the second quadrant, in which the origin is at the lower right. This allows the front and back origins to be aligned even if the exact media size is unknown.
<i>MaxCollect</i> ?	integer	<p>Maximum number of Sheets that will be collected into a signature. <i>@MaxCollect</i> modifies the pagination when automated imposition is selected.</p> <p>Specifying <i>@MaxCollect</i> can effectively cause a Page Pool or Page Pool List to be broken into “sub” Page Pools. Each of these “sub” Page Pools provides the set of pages mapped onto a single Collect, and are processed sequentially out of the “parent” Page Pool (or Page Pool List). Thus each sub-Page Pool effectively restarts the ord counting within the Imposition Template (i.e., treat a sub-Page Pool as if a new Page Pool were being started with the Imposition Template).</p> <p>If not specified, all sheets SHALL be collected.</p>
<i>MinCollect</i> ?	integer	Minimum number of Sheets that will be collected into a signature. <i>@MinCollect</i> modifies the pagination when automated imposition is selected.
<i>PaperRef</i> ?	IDREF	Reference to final paper Media . This Resource provides a description of the physical media which will be marked.
<i>PlateRef</i> ?	IDREF	Reference to plate Media . This Resource provides a description of the physical media which will be marked.

Table 8-115: Layout Resource (Sheet 3 of 5)

Name	Data Type	Description
<i>ProofPaperRef?</i>	IDREF	Reference to paper Media used for proofing. This Resource provides a description of the physical media which will be marked.
<i>SheetCountReset?</i>	enumeration	<p>Policy as to when the automated imposition variables <i>@SheetCount</i> and <i>@TotalSheetCount</i> are reset. See Section 6.4.18.2, “Variables for Automated Imposition” on page 377.</p> <p>Allowed values are:</p> <p><i>Continue</i> – <i>@SheetCount</i> continues to increment for each sheet generated by the current Imposition Template.</p> <p><i>PagePool</i> – <i>@SheetCount</i> is reset to zero upon start of processing of a new Page Pool and <i>@TotalSheetCount</i> is determined for that new Page Pool..</p> <p><i>PagePoolList</i> – <i>@SheetCount</i> is reset to zero upon start of processing of an Imposition Template, and <i>@TotalSheetCount</i> is recalculated. Note that the value of <i>@TotalSheetCount</i> may depend on the sheets generated from successive Imposition Templates (for example, if the current Imposition Template has <i>@SheetCountReset = "PagePoolList"</i>, and the subsequent Imposition Template has <i>@SheetCountReset = "Continue"</i>, <i>@TotalSheetCount</i> will include the sheets generated by both Imposition Templates).</p> <p>Note: <i>@SheetCount</i> and <i>@TotalSheetCount</i> are always reset to zero at the beginning of processing of a set regardless of the value of Layout/<i>@SheetCountReset</i>.</p>
<i>SheetNameFormat?</i>	string	<p>A formatting string used with <i>@SheetNameTemplate</i> to algorithmically construct <i>@SheetName</i>. <i>@SheetNameFormat</i> and <i>@SheetNameTemplate</i> are used to identify individual parts of the Layout in an automated environment.</p> <p>Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.</p>
<i>SheetNameTemplate?</i>	string	<p>A list of values used with <i>@SheetNameFormat</i> to algorithmically construct <i>@SheetName</i>. <i>@SheetNameFormat</i> and <i>@SheetNameTemplate</i> are used to identify individual parts of the Layout in an automated environment.</p> <p>Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.</p>
<i>SourceWorkStyle?</i>	WorkStyle	Indicates which <i>@WorkStyle</i> was used to create the Layout . This is only informative and can be useful when creating double sided proofs.

Table 8-115: Layout Resource (Sheet 4 of 5)

Name	Data Type	Description
<i>SurfaceContentsBox</i> ?	rectangle	<p>This box, specified in Layout-coordinate space, defines the area into which MarkObject or ContentObject Elements are distributed. The lower left corner of the rectangle specified by the value of this Attribute establishes the coordinate system into which the content is mapped and SHOULD have a value of "<i>0 0</i>".</p> <p><i>@SurfaceContentsBox</i> MAY imply clipping.</p> <p>This Attribute SHOULD be supplied in order to get predictable placement of content. If this Attribute is not supplied, a rectangle with the origin at "<i>0 0</i>" and an extent that MAY be dependent on the dimensions of one of the Media is implied.</p>
<i>TemplateType</i> ?	enumeration	<p>Specifies the type of automated Imposition Template being defined. If <i>@TemplateType</i> = "<i>ConditionalSheets</i>", then this Imposition Template SHALL only specify conditional sheet definitions (see Layout/SheetCondition). Typically, such an Imposition Template defines conditional sheets to be generated at the beginning and/or end of job and/or set. SHALL ONLY be specified if Layout/@Automated = "true".</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>ConditionalSheets</i> – the Imposition Template contains ONLY conditional sheet definitions <i>Normal</i> – the Imposition Template contains at least one sheet definition that consumes pages from the RunList (Document), and may contain conditional sheet definitions. If this value is specified in a partition leaf, this leaf SHALL NOT contain conditional sheet definitions.
<i>ContentObject</i> *	element	Provides a list of the ContentObject Elements to be placed on to the surface. Contains the marks on the surface in rendering order. All ContentObject Elements SHALL be specified Partition leaves of the Layout that contain at least one Part/@Surface .
<i>LogicalStackParams</i> ?	element	When specified, configures the imposition engine to place content onto one or more Logical Stacks distributed on a common set of sheets. Layout/LogicalStackParams SHALL only be specified in the root Layout element AND only when Layout/@Automated = "true". All Logical Stacks defined by LogicalStackParams SHALL be used in all Imposition Templates . See Section 6.4.18.4.1, "Using Logical Stacks" on page 382.
<i>MarkObject</i> *	element	Provides a list of the MarkObject Elements to be placed on to the surface. Contains the marks on the surface in rendering order. All MarkObject Elements SHALL be specified Partition leaves of the Layout that contain at least one Part/@Surface .

Table 8-115: Layout Resource (Sheet 5 of 5)

Name	Data Type	Description
PageCondition *	element	The PageCondition Elements are used only with automated imposition. They define restrictions on which page content may be placed in a Layout/ContentObject and. If any PageCondition restricts placing a page into a ContentObject , the Page SHALL NOT be filled into that ContentObject .
Position *	element	The Position Element specifies how the BinderySignature is placed onto a Sheet. Multiple Position objects that reference the same BinderySignature specify multiple identical BinderySignature Elements with the same content. In case the BinderySignature is defined by Signature-Cells, then, by default, the front pages are placed on the front side of the Sheet and the back pages are placed on the back side of the Sheet. Using the @Orientation Attribute one can influence this default behavior. When the BinderySignature is defined by @FoldCatalog or Fold Elements, then, by default, the day is placed on the left front side of the Sheet. Using the @Orientation Attribute one can influence this default behavior. Position SHALL NOT be specified at Partition levels lower than @PartVersion .
SheetCondition ?	element	Specifies the conditions under which the optional sheet defined by this Layout is produced. SHALL only be present when Layout/@Automated = "true" , and SHALL be contained within a Layout branch partitioned by @SheetName .

8.60.1 Element: ColorControlStrip

This Resource describes a color control strip. The type of the color control strip is given in the **@StripType** Attribute. The lower left corner of the control strip box is used as the origin of the coordinate system used for the definition of the measuring fields. It can be calculated using the following formula:

$$x_0 = x - \frac{w}{2} \cos(\varphi) + \frac{h}{2} \sin(\varphi)$$

$$y_0 = y - \frac{w}{2} \sin(\varphi) - \frac{h}{2} \cos(\varphi)$$

where

x = X element of the **@Center** Attribute

y = Y element of the **@Center** Attribute

w = X element of the **@Size** Attribute

h = Y element of the **@Size** Attribute

j = Value of the **@Rotation** Attribute

Table 8-116: ColorControlStrip Element (Sheet 1 of 2)

Name	Data Type	Description
Center ?	XYPair	Position of the center of the color control strip in the coordinates of the MarkObject that contains this mark.

Table 8-116: ColorControlStrip Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Rotation</i> ?	double	Rotation in degrees. Positive graduation figures indicate counter-clockwise rotation; negative figures indicate clockwise rotation.
<i>Separations</i> ?	NMTOKENS	Ordered list of separations that comprise the ColorControlStrip . If neither CIELABMeasuringField nor DensityMeasuringField are specified, the geometry is implied by the value of @StripType .
<i>Size</i> ?	XYPair	Size, in points, of the color control strip.
<i>StripType</i> ? <u>Modified</u>	string	Type of color control strip. This Attribute MAY be used for specifying a predefined, company-specific color control strip.
CIELABMeasuringField *	element	Details of a CIELAB measuring field that is part of this ColorControlStrip .
DensityMeasuringField *	element	Details of a density measuring field that is part of this ColorControlStrip .

8.60.2 Element: CIELABMeasuringField

Information about a color measuring field. The color is specified as CIE-L*a*b* value.

Table 8-117: CIELABMeasuringField Element

Name	Data Type	Description
<i>Center</i>	XYPair	Position of the center of the color measuring field in the coordinates of the MarkObject that contains this mark. If the measuring field is defined within a ColorControlStrip , @Center refers to the rectangle defined by @Center and @Size of the ColorControlStrip .
<i>CIELab</i>	LabColor	L*a*b* color specification.
<i>Diameter</i> ?	double	Diameter of the measuring field.
<i>Percentages</i> ?	DoubleList	Percentage values for each separation. The number of array Elements SHALL match the number of separations.
<i>ScreenRuling</i> ?	DoubleList	Screen ruling values in lines per inch for each separation. The number of array Elements SHALL match the number of separations.
<i>ScreenShape</i> ?	string	Shape of screening dots.
<i>Tolerance</i> ?	double	Tolerance in ΔE.

8.60.3 Element: DensityMeasuringField

This contains information about a density measuring field.

Table 8-118: DensityMeasuringField Resource

Name	Data Type	Description
<i>Center</i>	XYPair	Position of the center of the density measuring field in the coordinates of the MarkObject that contains this mark. If the measuring field is defined within a ColorControlStrip , <i>@Center</i> refers to the rectangle defined by <i>@Center</i> and <i>@Size</i> of the ColorControlStrip .
<i>Density</i>	DoubleList	Density value for each process color measured with filter. The sequence of colors is C M Y K, as in the data type CMYKColor.
<i>Diameter</i>	double	Diameter of measuring field.
<i>DotGain</i>	double	Percentage of dot gain.
<i>Percentage</i>	double	Film percentage or equivalent.
<i>Screen</i>	string	Description of the screen.
<i>Separation</i>	string	Reference to a Separation that this applies DensityMeasuringField to.
<i>Setup ?</i>	string	Description of measurement setup.
<i>ToleranceBlack</i>	XYPair	Upper and lower black measurement limits (in density units).
<i>ToleranceCyan</i>	XYPair	Upper and lower cyan measurement limits (in density units).
<i>ToleranceDotGain</i>	XYPair	Upper and lower measurement limits (in%).
<i>ToleranceMagenta</i>	XYPair	Upper and lower magenta measurement limits (in density units).
<i>ToleranceYellow</i>	XYPair	Upper and lower yellow measurement limits (in density units).

8.60.4 Element: DeviceMark

The **DeviceMark** Element specifies the formatting parameters for how text for a device mark should be marked. This text is provided by an associated **JobField** Element (see **Layout/MarkObject/JobField** or **LayoutElementProductionParams/JobField**).

Two methods for text layout are provided by **DeviceMark**. First, text can be placed within a bounding box defined by a containing **MarkObject** (see **MarkObject/@TrimSize** for defining the size of that bounding box). When this feature is selected, **DeviceMark/@Font**, **DeviceMark/@FontSize**, **DeviceMark/@HorizontalFitPolicy** and **DeviceMark/@VerticalFitPolicy** MAY be used to specify how text SHALL be fit within that bounding box.

The second method allows the bounding box defined by the text itself to be positioned, rotated, and scaled (along with the text). This facility operates through specifying an anchor point on that bounding box, and having the **MarkObject/@CTM** operate relative to that anchor point. **DeviceMark** Attributes that affect this method are **DeviceMark/@Font** and **DeviceMark/@FontSize**.

See figures below for illustrations of marks generated by **DeviceMark**.

Element Properties

Element referenced by: Layout/MarkObject

Table 8-119: DeviceMark Element

Name	Data Type	Description
<i>Anchor?</i>	Anchor	Anchor point on or within the bounding box of the text marked by this DeviceMark that MarkObject/@CTM refers to. When @Anchor is specified, MarkObject/@TrimSize, DeviceMark/@HorizontalFitPolicy and DeviceMark/@VerticalFitPolicy are ignored. Note: the bounding box of this DeviceMark is defined by the extent of the text being marked.
<i>Font?</i>	NMTOKEN	The name of the font that is to be used for the DeviceMark. Values include: <i>Courier</i> <i>Helvetica</i> <i>Helvetica-Condensed</i> <i>Times-Roman</i>
<i>FontSize?</i>	double	The size of the font that is to be used for the DeviceMark, in points ≥ 0 .
<i>HorizontalFitPolicy?</i>	enumeration	Allowed values are from: StripMark/@HorizontalFitPolicy.
<i>VerticalFitPolicy?</i>	enumeration	Allowed values are from: StripMark/@VerticalFitPolicy
<i>BarcodeReproParams?</i>	element	Reproduction parameters for Barcodes specified in the parent MarkObject/ IdentificationField .

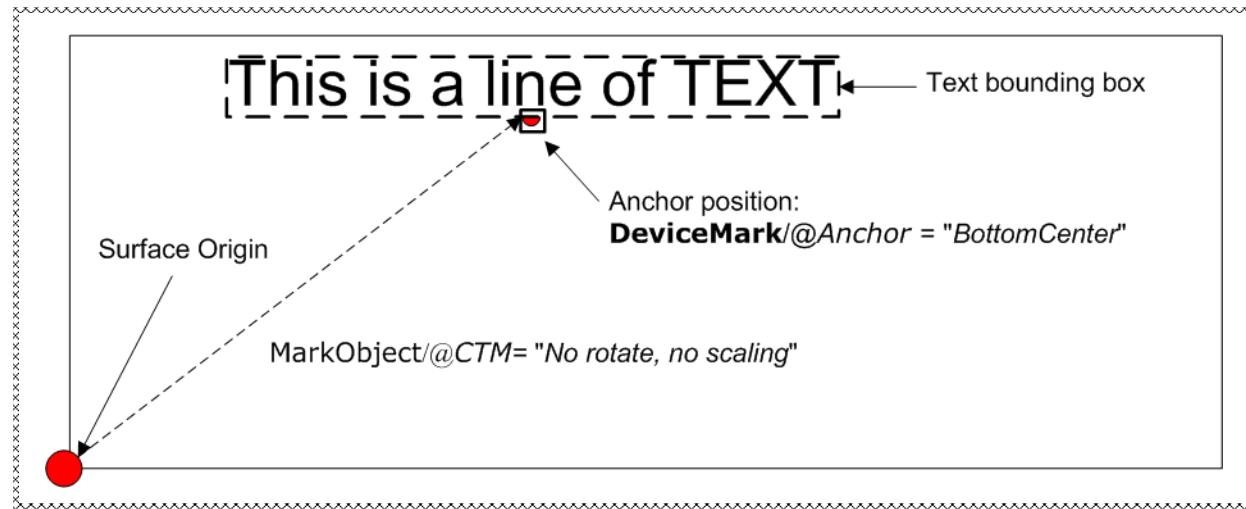
Figure 8-34: Anchor with No Scaling and No Rotation

Figure 8-35: Anchor with No Scaling and Rotation of 90° Clockwise

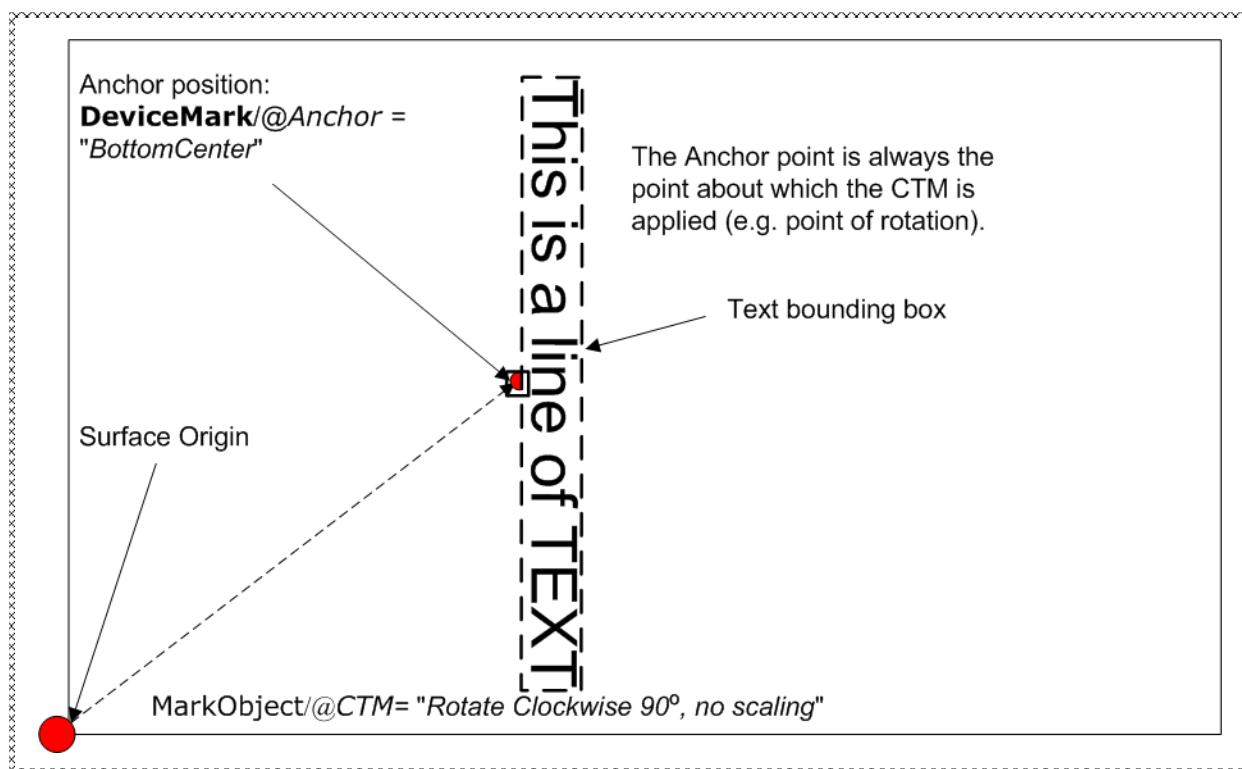
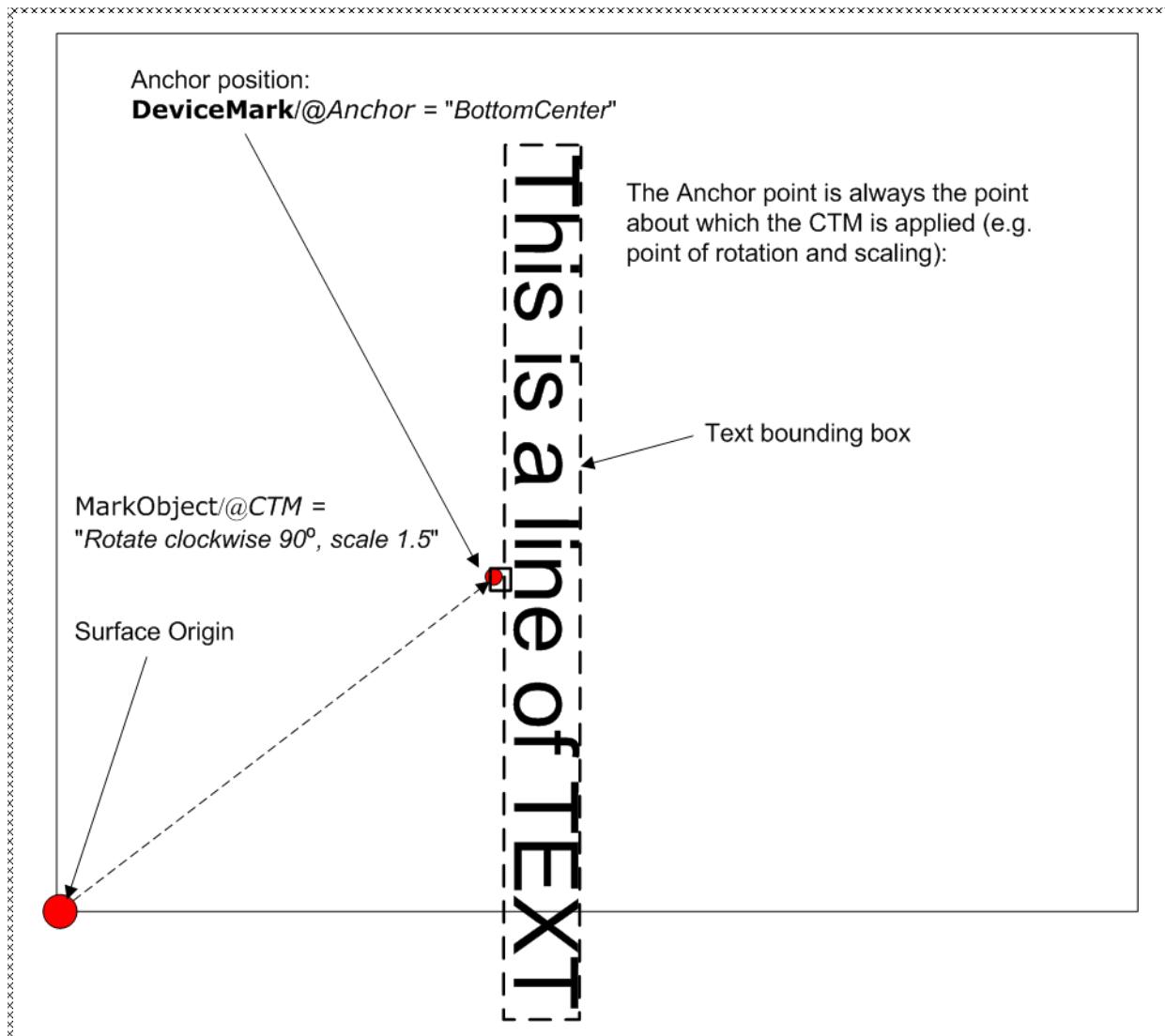


Figure 8-36: Anchor with 1.5 Scaling and Rotation of 90° Clockwise

8.60.5 Element: LogicalStackParams

Table 8-120: LogicalStackParams Element (Sheet 1 of 2)

Name	Data Type	Description
<i>MaxStackDepth</i> ?	integer	Maximum number of imposed sheets to generate as an Imposed Sheet Set (the size of the Logical Stack). Implementations SHALL generate the minimum stack size to accommodate the available number of Logical Sheets if the total number of required sheets for the last stack is smaller than <i>@MaxStackDepth</i> .

Table 8-120: LogicalStackParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Restrictions</i> ?	enumeration	<p>Describes any restrictions set on the placement of a Recipient Set's Logical Sheets within or across Imposed Sheet Sets.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>None</i> – a recipient set's Logical Sheets may be placed across both Logical Stacks and Imposed Sheet Sets. <i>WithinImposedSheetSet</i> – a Recipient Set's Logical Sheets SHALL be placed within a single Imposed Sheet Set <i>WithinLogicalStack</i> – a Recipient Set's Logical Sheets SHALL be placed within a single Logical Stack.
Stack *	element	Describes parameters to control the sequencing of Logical Sheets onto individual Logical Stacks.

8.60.6 Element: Stack

Table 8-121: Stack Element

Name	Data Type	Description
<i>LogicalStackOrd</i>	integer	0-based Logical Stack identifier that specifies which Logical Stack is controlled by this Stack Element. The value of Stack/@ <i>LogicalStackOrd</i> SHALL correspond to a ContentObject/@ <i>LogicalStackOrd</i> or MarkObject/@ <i>LogicalStackOrd</i> value.
<i>LogicalStackSequence</i> ?	enumeration	<p>Specifies how Logical Sheets SHALL be placed onto the Logical Stack.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>SheetIndex</i> – Logical Sheets are placed in the order of ascending @<i>SheetIndex</i>. <i>DescendingSheetIndex</i> – Logical Sheets are placed in the order of descending @<i>SheetIndex</i>.

8.60.7 Element: PageCondition

The PageCondition Element defines restrictions on when page content SHALL NOT be placed in a ContentObject of a Layout. Before placing page content from a RunList into a ContentObject the PageCondition/@*RestrictedContentObjects* Attribute SHALL be checked for the @*Ord* of the ContentObject. If the @*Ord* of the ContentObject is in the @*RestrictedContentObjects* Attribute Value, the alternate content, if any, SHALL be placed in the ContentObject. After skipping a restricted ContentObject, the **Imposition** Process SHALL then place the current page content into the location defined by the next ContentObject (after that specified by the @*RestrictedContentObject*). This corresponds to incrementing the effective @*Ord* value of the page in the RunList by 1, effectively incrementing the total number of pages of the RunList. If the next ContentObject is also restricted then the process is repeated. PageCondition Elements are processed in their XML order.

Table 8-122: PageCondition Element

Name	Data Type	Description
<i>Condition</i> ?	enumeration	<p>Specifies the conditions when the <i>PageCondition</i> applies. Condition SHALL NOT be specified if <i>Part</i> Elements are present.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>PagePoolStart</i> – the condition is true when the <i>@Ord</i> refers to the first page of a Page Pool in the RunList. <i>PagePoolEnd</i> – after processing of the Page Pool is completed, the condition is true for all unused <i>@Ord</i> positions in the current Collect. <i>PagePoolListStart</i> – the condition is true when the <i>@Ord</i> refers to the first page of an aggregated set of Page Pools in the RunList. <i>PagePoolListEnd</i> – after processing of the Page Pool list is completed, the condition is true for all unused Ord positions in the current Collect.
<i>RestrictedContentObjects</i>	IntegerList	List of <i>@Ord</i> values of those <i>ContentObject</i> Elements into which page content that matches the conditions as specified in <i>Part</i> or <i>PageCondition/@Condition</i> SHALL NOT be placed.
<i>RunListRef</i> ?	IDREF	<p>Alternate page content that SHALL be placed into the <i>ContentObject</i> Elements that are specified in <i>@RestrictedContentObjects</i> when the <i>PageCondition</i> evaluates to "true". The first page of the referenced RunList SHALL be used.</p> <p>Note: the behavior of providing alternate content using RunList is defined only if <i>@Condition</i> is specified.</p>
<i>Part</i> *	element	<p>Specifies the conditions when the <i>PageCondition</i> applies. Multiple <i>Part</i> Elements specify alternate page conditions (ORing of them).</p> <p><i>Part</i> Elements SHALL NOT be specified if <i>@Condition</i> is present.</p>

Example 8-13: PageCondition

TBD 2.x Example.

```

<Layout Class="Parameter" ID="L0000004" Status="Available"
        PartIDKeys="SheetName Side" BaseOrdReset="PagePoolList">
  <PageCondition RestrictedContentObjects="1 -1">
    <!--
      This example assumes that the pages of a sequence of documents of the
      RunList are to be treated as an aggregate page pool, and the pages are
      to be saddle stitch imposed onto a continuous sequence of sheets. Some
      documents of the sequence represent a start of a new chapter where their
      DocTag is set to the value 'Chapter'. These chapter starts force the
      first page of each chapter to be placed on the right side finished page.
    -->
    <Part DocTags="Chapter" DocRunIndex="0"/>
  </PageCondition>
<Layout SheetName="Mysheet">
  <Layout Side="Front">

```

```
<ContentObject CTM="1 0 0 1 0 0" Ord="-1"/>    <!-- Outside left -->
<ContentObject CTM="1 0 0 1 595 0" Ord="0" /> <!-- outside right -->
</Layout>
<Layout Side="Back">
    <ContentObject CTM="1 0 0 1 0 0" Ord="1"/>    <!-- inside left-->
    <ContentObject CTM="1 0 0 1 595 0" Ord="-2"/> <!-- inside right-->
</Layout>
</Layout>
</Layout>
```

8.60.8 Element: SheetCondition

Table 8-123: SheetCondition Element

Name	Data Type	Description
<i>Condition</i> ?	enumerations	<p>When present, defines an optional sheet by specifying each condition (equivalent to a logical or) under which the optional sheet is produced.</p> <p>Values include:</p> <p><i>Begin</i> – At beginning of imposition processing (all sets in case of multiple Recipient Sets)</p> <p><i>End</i> – At the end of imposition processing.</p> <p><i>BeginSet</i> – At beginning of processing of an individual Recipient Set.</p> <p><i>EndSet</i> – At end of processing of an individual Recipient Set.</p> <p><i>PagePoolBegin</i> – At beginning of processing of a Page Pool.</p> <p><i>PagePoolEnd</i> – At the end of processing of a Page Pool.</p> <p><i>PagePoolListBegin</i> – At beginning of processing of a Page Pool List</p> <p><i>PagePoolListEnd</i> – At end of processing of a Page Pool List</p> <p><i>LogicalStackBegin</i> – adds a Logical Sheet to the beginning of each Logical Stack generated as part of an Imposed Sheet Set.</p> <p><i>LogicalStackEnd</i> – adds a Logical Sheet to the end of each Logical Stack generated as part of an Imposed Sheet Set.</p> <p><i>LogicalStackSetBegin</i> – At beginning of generation of a set of Logical Stacks. Note that <i>@LogicalStackOrd</i> SHALL be used to indicate the Logical Stack on which the conditional sheet is placed.</p> <p><i>LogicalStackSetEnd</i> – At end of generation of a set of Logical Stacks. Note that <i>@LogicalStackOrd</i> SHALL be used to indicate the Logical Stack on which the conditional sheet is placed.</p> <p><i>ImposedSheetSetBegin</i> – At beginning of generation of an Imposed Sheet Set. Note that this generates a separator sheet not counted as part of the Imposed Sheet Set. Any MarkObject Elements specifying <i>@LogicalStackOrd</i> are ignored.</p> <p><i>ImposedSheetSetEnd</i> – At end of generation of an Imposed Sheet Set. Note that this generates a separator sheet not counted as part of the Imposed Sheet Set. Any MarkObject Elements specifying <i>@LogicalStackOrd</i> are ignored.</p>
<i>RunListRef</i> ?	IDREF	Supplies content for any ContentObject Elements specified within the optional sheet definition. All ContentObject Elements in the optional sheet definition SHALL reference content supplied by this RunList .

8.60.9 PlacedObject

MarkObject elements and ContentObject Elements describe the placement of content onto a sheet surface.

8.60.10 Element: ContentObject

ContentObject Elements describe containers for page content on a surface. They are filled from the content RunList of the **Imposition** Process. For print applications where page count varies from Instance Document to Instance Document, imposition templates can automatically assign pages to the correct surface and ContentObject position.

Table 8-124: ContentObject Element (Sheet 1 of 5)

Name	Data Type	Description
<i>Anchor</i> ?	Anchor	<p>Specifies the anchor point of the ContentObject that remains in place on the surface when the value of <i>@TrimSize</i> changes. <i>@Anchor</i> is specified in the coordinate system of the ContentObject prior to application of the <i>@CTM</i>.</p> <p>Note: the <i>@Anchor</i> Attribute is metadata used to identify to an Imposition generation utility a fixed anchor point reference to an abstract content page. This may occur when an XJDF Layout Resource is used as a template for that utility. This attribute has no effect on processing when a Layout Resource is input to the Imposition Process.</p>
<i>BinderySignatureIDs</i> ?	NMTOKENS	<p>Identification of the Assembly Elements if Stripping describes an imposition scheme for multiple Assembly Elements. <i>@BinderySignatureIDs</i> MAY contain multiple NMTOKENS, when the Assembly Resource specifies an intermediate product that contains multiple final assemblies or when multiple BinderySignature elements are bound into one Assembly. If specified, <i>@BinderySignatureIDs</i> SHALL list the BinderySignature/<i>@BinderySignatureID</i> of all BinderySignatures that contain pages that are referenced by this ContentObject.</p>
<i>ClipBox</i> ?	rectangle	<p>Clipping rectangle in the coordinates of the <i>@SurfaceContentsBox</i>. <i>@ClipBox</i> SHALL NOT be present if ContentObject/<i>@ClipBoxFormat</i> is supplied.</p>
<i>ClipBoxFormat</i> ?	string	<p>A formatting string used with <i>@ClipBoxTemplate</i> to algorithmically construct <i>@ClipBox</i>. <i>@ClipBoxFormat</i> SHALL ONLY be present if ContentObject/<i>@ClipBox</i> is not supplied and Layout/<i>@Automated</i> = "true".</p> <p>Allowed values are from: Appendix I, "Generating strings with Format and Template" on page 1203.</p>
<i>ClipBoxTemplate</i> ?	string	<p>A list of values used with <i>@ClipBoxFormat</i> to algorithmically construct <i>@ClipBox</i>. <i>@ClipBoxTemplate</i> SHALL ONLY be present if ContentObject/<i>@ClipBox</i> is not supplied and Layout/<i>@Automated</i> = "true".</p> <p>Allowed values are from: Appendix I, "Generating strings with Format and Template" on page 1203.</p>

Table 8-124: ContentObject Element (Sheet 2 of 5)

Name	Data Type	Description
<i>ClipPath?</i>	PDFPath	Clip path for the <i>ContentObject</i> in the coordinates of the <i>@SurfaceContentsBox</i> (lower left of <i>@SurfaceContentsBox</i> is used as reference zero point, same as for <i>@ClipBox</i>). The actual clip region is the intersection of <i>@ClipBox</i> and <i>@ClipPath</i> or the intersection of <i>@ClipBox</i> and <i>@SourceClipPath</i> . Thus both clip paths are applied sequentially and the resulting clip region is smaller than each individual clip Box or Path. <i>@ClipPath</i> and <i>@SourceClipPath</i> SHALL NOT be specified in the same <i>ContentObject</i> . <i>@ClipPath</i> SHOULD be specified when both <i>@ClipPath</i> and <i>@SourceClipPath</i> are known because <i>@ClipPath</i> provides a more stable coordinate system (not sensitive to shifts caused by editing the page).
<i>CompensationCTMFormat?</i>	string	A formatting string used with <i>@CompensationCTMTemplate</i> to algorithmically construct a compensation CTM that SHALL be concatenated to CTM. <i>@CompensationCTMFormat</i> MAY be present if <i>Layout/@Automated = "true"</i> . Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.
<i>CompensationCTMTemplate?</i>	string	A list of values used with <i>@CompensationCTMFormat</i> to algorithmically construct a compensation CTM that SHALL be concatenated to CTM. <i>@CompensationCTMTemplate</i> MAY be present if <i>Layout/@Automated = "true"</i> . Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.
<i>CTM</i>	matrix	The coordinate transformation matrix (CTM — a Postscript term) of the object in the <i>@SurfaceContentsBox</i> . For details, see Figure 2-10, “Equation for Surface Coordinate System Transformations,” on page 41. The origin of the source coordinate system is the lower left (expressed in the source coordinate system) of the object and the origin of the destination coordinate system is lower left of the <i>@SurfaceContentsBox</i> . For details, see Section 2.5.1.1, “Source Coordinate Systems” on page 37. Note: <i>@CTM</i> SHALL be recalculated if the object is replaced afterwards with a new object with different dimensions.
<i>DocOrd ?</i>	integer	Reference to an index of an Instance Document in the content RunList . This references an Instance Document with an index module. <i>Layout/@MaxDocOrd</i> equals <i>@DocOrd</i> in an automated layout scenario. The index can either be known explicitly from a variable RunList or implicitly from the index within an indexable content definition language (e.g., PPML).

Table 8-124: ContentObject Element (Sheet 3 of 5)

Name	Data Type	Description
<i>FitPolicyRef</i> ?	IDREF	SHALL NOT be present when Layout/@Automated = "false" . Specifies automated fit policy for the page cell described by the ContentObject . When present, ContentObject/@TrimSize SHALL also be present in the ContentObject , and represents the cell size for this ContentObject .
<i>HalfTonePhaseOrigin</i> ?	XYPair	Location of the origin for screening of this ContentObject . Specified in the coordinate systems of @SurfaceContentsBox .
<i>ID</i> ?	ID	Identifier for referencing this ContentObject from MarkObject/@ContentRef .
<i>LogicalStackOrd</i> ?	integer	0-based Logical Stack identifier that this ContentObject belongs to. @LogicalStackOrd SHALL match the @LogicalStackOrd of an entry in Layout/LogicalStackParams/Stack .
<i>Ord</i> ?	integer	A zero-based reference to an index in the content RunList . The index is incremented for every page of the RunList with @IsPage = "true" . The @Ord value of the first page of a RunList has the value "0" . If Layout/@Automated = "true" , @Ord MAY be a negative integer in a ContentObject . In this case, the explicit @Ord for each iteration of the automated Layout is calculated by subtracting the appropriate number of @Ord values from the back of the document. For details on automated Layout , see Section 6.4.18, “ Imposition ” on page 372.
<i>OrdExpression</i> ?	string	Function to calculate an @Ord value dynamically, using a value of s for Signature number and n for total number of pages in the Instance Document. The @Ord or @DocOrd and @OrdExpression are mutually exclusive in one ContentObject . Value format is from: Section 8.87.19.5, “ Using Expressions in the OrdExpression Attribute ” on page 762.
<i>SetOrd</i> ?	integer	A non-negative, zero-based reference to an index of a Document Set in the content RunList . This references an Instance Document with an index module. Layout/@MaxSetOrd = @SetOrd in an automated layout scenario. The index can either be known explicitly from a variable RunList or implicitly from the index within an indexable content definition language (e.g., PPML).

Table 8-124: ContentObject Element (Sheet 4 of 5)

Name	Data Type	Description
<i>SourceClipPath</i> ?	PDFPath	<p>Clip path for the <i>ContentObject</i> in the source coordinate system. <i>@SourceClipPath</i> is applied to the referenced source object in addition to any clipping that is internal to the object. Internal transformation of the source object (Rotation key in PDF, Orientation Tag in TIFF etc.) SHALL be applied prior to applying <i>@SourceClipPath</i>.</p> <p><i>@ClipPath</i> and <i>@SourceClipPath</i> SHALL NOT be specified in the same <i>ContentObject</i>.</p> <p>See Section 2.5.1.1, “Source Coordinate Systems” for definitions of source coordinate systems.</p>
<i>TrimClipPath</i> ?	PDFPath	<p>The die cutting path for the <i>ContentObject</i> in the coordinates of the <i>@SurfaceContentsBox</i> (lower left of <i>@SurfaceContentsBox</i> is used as reference zero point, same as for <i>@ClipBox</i>). That path can be used for proofing purpose.</p> <p>Note: the <i>@TrimClipPath</i> Attribute may be used by an Imposition generation utility when an XJDF Layout Resource is used as a template for that utility. This Attribute has no effect on processing when a Layout Resource is input to the Imposition Process.</p>
<i>TrimCTM</i> ?	matrix	<p>The transformation matrix of the trim box to be applied to the object’s referenced content in the coordinate system of <i>@SurfaceContentsBox</i>. Note that imposition programs that execute the Layout SHALL recalculate the <i>@CTM</i> in case the referenced content is replaced with new referenced content having different dimensions, otherwise the position of the content inside the trim box will shift. This recalculation is based on <i>@Anchor</i>, <i>@TrimCTM</i>, <i>@TrimSize</i> and trim box.</p> <p>Note: the <i>@TrimCTM</i> Attribute may be used by an Imposition generation utility when an XJDF Layout Resource is used as a template for that utility. This Attribute has no effect on processing when a Layout Resource is input to the Imposition Process.</p>

Table 8-124: ContentObject Element (Sheet 5 of 5)

Name	Data Type	Description
<i>TrimSize?</i>	XYPair	<p>The size of the object's trim box as viewed in the object source coordinates ((@<i>TrimCTM</i> scaling and rotation NOT applied)).</p> <p>@<i>TrimSize</i> is needed when replacing the object by a new object with a different dimension.</p> <p>When a Layout Resource is input to the Imposition Process, @<i>TrimSize</i> specifies the bounding box to be used for text layout when processing a MarkObject/DeviceMark or for scaling and rotation when processing ContentObject/FitPolicy.</p> <p>Note: Recalculation of ContentObject/@CTM is only necessary when the Stripping Process or application needs to replace some pages from the provided RunList (using the Layout as a kind of imposition “template”). To ensure correct placement of a new page in the Layout, ContentObject/@CTM recalculations SHOULD always be done according to ContentObject/@TrimCTM and ContentObject/@TrimSize. Together, these two Attributes represent the trimming information of the imposition software page, which is not always the same as the original RunList page trimming information (= RunList/@SourceTrimBox when real trim box of the object is known).</p> <p>Usage of both ContentObject Elements @<i>TrimCTM</i> and @<i>TrimSize</i> Attributes will allow page replacements on any type of imposition Layout.</p>

8.60.11 Element: **MarkObject**

MarkObject Elements describe containers for production marks on a surface. The PDL for the marks MAY exist prior to imposing. In this case The Marks SHOULD be filled from the **RunList** (**Marks**) of the **Imposition** Process. Marks MAY instead be specified in a manner that allows the imposition engine to generate the marks on the fly. An individual **MarkObject** represents the content data of the Marks. The content data in individual **MarkObject** Elements MAY contain multiple logical marks.

Table 8-125: MarkObject Element (Sheet 1 of 6)

Name	Data Type	Description
<i>Anchor?</i>	Anchor	<p>Specifies the anchor point of the MarkObject that remains in place on the surface when the value of @<i>TrimSize</i> changes. @<i>Anchor</i> is specified in the coordinate system of the MarkObject prior to application of the @<i>CTM</i>.</p> <p>Note: the @<i>Anchor</i> Attribute is metadata used to identify to an Imposition generation utility a fixed anchor point reference to an abstract content page. This may occur when an XJDF Layout Resource is used as a template for that utility. This attribute has no effect on processing when a Layout Resource is input to the Imposition Process.</p>

Table 8-125: **MarkObject** Element (Sheet 2 of 6)

Name	Data Type	Description
<i>BinderySignatureIDs?</i>	NMTOKENS	Identification of the Assembly Elements if <i>Stripping</i> describes an imposition scheme for multiple Assembly Elements. @ <i>BinderySignatureIDs</i> MAY contain multiple NMTOKENS, when the Assembly Resource specifies an intermediate product that contains multiple final assemblies or when multiple BinderySignature elements are bound into one Assembly . If specified, @ <i>BinderySignatureIDs</i> SHALL list the BinderySignature /@ <i>BinderySignatureID</i> of all BinderySignatures that contain pages that are referenced by this MarkObject .
<i>ClipBox?</i>	rectangle	Clipping rectangle in the coordinates of the @ <i>SurfaceContentsBox</i> . @ <i>ClipBox</i> SHALL NOT be present if MarkObject /@ <i>ClipBoxFormat</i> is supplied.
<i>ClipBoxFormat?</i>	string	A formatting string used with @ <i>ClipBoxTemplate</i> to algorithmically construct @ <i>ClipBox</i> . @ <i>ClipBoxFormat</i> SHALL ONLY be present if MarkObject /@ <i>ClipBox</i> is not supplied and Layout /@ <i>Automated</i> = "true". Allowed values are from: Appendix I, "Generating strings with Format and Template" on page 1203.
<i>ClipBoxTemplate?</i>	string	A list of values used with @ <i>ClipBoxFormat</i> to algorithmically construct @ <i>ClipBox</i> . @ <i>ClipBoxTemplate</i> SHALL ONLY be present if MarkObject /@ <i>ClipBox</i> is not supplied and Layout /@ <i>Automated</i> = "true". Allowed values are from: Appendix I, "Generating strings with Format and Template" on page 1203.
<i>ClipPath?</i>	PDFPath	Clip path for the MarkObject in the coordinates of the @ <i>SurfaceContentsBox</i> (lower left of @ <i>SurfaceContentsBox</i> is used as reference zero point, same as for @ <i>ClipBox</i>). The actual clip region is the intersection of @ <i>ClipBox</i> and @ <i>ClipPath</i> or the intersection of @ <i>ClipBox</i> and @ <i>SourceClipPath</i> . Thus both clip paths are applied sequentially and the resulting clip region is smaller than each individual clip Box or Path. @ <i>ClipPath</i> and @ <i>SourceClipPath</i> SHALL NOT be specified in the same MarkObject . @ <i>ClipPath</i> SHOULD be specified when both @ <i>ClipPath</i> and @ <i>SourceClipPath</i> are known because @ <i>ClipPath</i> provides a more stable coordinate system (not sensitive to shifts caused by editing the page).
<i>CompensationCTMFormat?</i>	string	A formatting string used with @ <i>CompensationCTMTemplate</i> to algorithmically construct a compensation CTM that SHALL be concatenated to CTM. @ <i>CompensationCTMFormat</i> MAY be present if Layout /@ <i>Automated</i> = "true". Allowed values are from: Appendix I, "Generating strings with Format and Template" on page 1203.

Table 8-125: MarkObject Element (Sheet 3 of 6)

Name	Data Type	Description
<i>CompensationCTMTemplate</i> ?	string	A list of values used with <i>@CompensationCTMFormat</i> to algorithmically construct a compensation CTM that SHALL be concatenated to CTM. <i>@CompensationCTMTemplate</i> MAY be present if Layout/@Automated = "true" . Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.
<i>ContentRef</i> ?	IDREF	<i>@ContentRef</i> refers to the ContentObject that this MarkObject is related to. <i>@ContentRef</i> is used to define the object that metadata for generating dynamic marks MAY be extracted from.
<i>CTM</i> ?	matrix	The coordinate transformation matrix (CTM — a Postscript term) of the object in the <i>@SurfaceContentsBox</i> . For details, see Figure 2-10, “Equation for Surface Coordinate System Transformations,” on page 41. The origin of the source coordinate system is the lower left (expressed in the source coordinate system) of the object and the origin of the destination coordinate system is lower left of the <i>@SurfaceContentsBox</i> . For details, see Section 2.5.1.1, “Source Coordinate Systems” on page 37. Note: <i>@CTM</i> SHALL be recalculated if the object is replaced afterwards with a new object with different dimensions.
<i>HalfTonePhaseOrigin</i> ?	XYPair	Location of the origin for screening of this ContentObject . Specified in the coordinate systems of <i>@SurfaceContentsBox</i> .
<i>LogicalStackOrd</i> ?	integer	0-based Logical Stack identifier that this MarkObject belongs to. <i>@LogicalStackOrd</i> SHALL match the <i>@LogicalStackOrd</i> of an entry in Layout/LogicalStackParams/Stack .
<i>Ord</i> ?	integer	A non-negative reference to an index in the RunList (Marks) . The index is incremented for every page of the RunList with <i>@IsPage = "true"</i> . The first page of a RunList has the value 0.
<i>SourceClipPath</i> ?	PDFPath	Clip path for the MarkObject in the source coordinate system. <i>@SourceClipPath</i> is applied to the referenced source object in addition to any clipping that is internal to the object. Internal transformation of the source object (Rotation key in PDF, Orientation Tag in TIFF etc.) SHALL be applied prior to applying <i>@SourceClipPath</i> . <i>@ClipPath</i> and <i>@SourceClipPath</i> SHALL NOT be specified in the same MarkObject . See Section 2.5.1.1, “Source Coordinate Systems” for definitions of source coordinate systems.

Table 8-125: MarkObject Element (Sheet 4 of 6)

Name	Data Type	Description
<i>TrimClipPath?</i>	PDFPath	<p>The die cutting path for the MarkObject in the coordinates of the @SurfaceContentsBox (lower left of @SurfaceContentsBox is used as reference zero point, same as for @ClipBox). That path can be used for proofing purpose.</p> <p>Note: the @TrimClipPath Attribute may be used by an Imposition generation utility when an XJDF Layout Resource is used as a template for that utility. This Attribute has no effect on processing when a Layout Resource is input to the Imposition Process.</p>
<i>TrimCTM?</i>	matrix	<p>The transformation matrix of the trim box to be applied to the object's referenced content in the coordinate system of @SurfaceContentsBox. Note that imposition programs that execute the Layout SHALL recalculate the @CTM in case the referenced content is replaced with new referenced content having different dimensions, otherwise the position of the content inside the trim box will shift. This recalculation is based on @Anchor, @TrimCTM, @TrimSize and trim box.</p> <p>Note: the @TrimCTM Attribute may be used by an Imposition generation utility when an XJDF Layout Resource is used as a template for that utility. This Attribute has no effect on processing when a Layout Resource is input to the Imposition Process.</p>

Table 8-125: MarkObject Element (Sheet 5 of 6)

Name	Data Type	Description
<i>TrimSize</i> ?	XYPair	<p>The size of the object's trim box as viewed in the object source coordinates ((@<i>TrimCTM</i> scaling and rotation NOT applied)).</p> <p>@<i>TrimSize</i> is needed when replacing the object by a new object with a different dimension.</p> <p>When a Layout Resource is input to the Imposition Process, @<i>TrimSize</i> specifies the bounding box to be used for text layout when processing a MarkObject/DeviceMark or for scaling and rotation when processing MarkObject/FitPolicy.</p> <p>Note: Recalculation of MarkObject/@CTM is only necessary when the Stripping Process or application needs to replace some pages from the provided RunList (using the Layout as a kind of imposition "template"). To ensure correct placement of a new page in the Layout, MarkObject/@CTM recalculations SHOULD always be done according to MarkObject/@TrimCTM and MarkObject/@TrimSize. Together, these two Attributes represent the trimming information of the imposition software page, which is not always the same as the original RunList page trimming information (= RunList/@SourceTrimBox when real trim box of the object is known).</p> <p>Usage of both PlacedObject Elements @TrimCTM and @TrimSize Attributes will allow page replacements on any type of imposition Layout.</p>
CIELABMeasuringField *	element	Specific information about this kind of mark object. See below for information regarding dynamically generated marks.
ColorControlStrip *	element	Specific information about this kind of mark object. See below for information regarding dynamically generated marks.
CutMark *	element	Specific information about this kind of mark object. See below for information regarding dynamically generated marks.
DensityMeasuringField *	element	Specific information about this kind of mark object. See below for information regarding dynamically generated marks.
DeviceMark ?	element	<p>DeviceMark specifies all formatting options for generating dynamic marks that are generated on the fly.</p> <p>Constraint: at most one of @<i>Ord</i> or DeviceMark SHALL be specified.</p>
DynamicField *	element	Definition of text replacement for this MarkObject . MarkObject/DynamicField specifies text replacement within an existing PDL mark.
FillMark *	element	Specifies marks that define a fill layer, e.g., for backlit displays.

Table 8-125: MarkObject Element (Sheet 6 of 6)

Name	Data Type	Description
FitPolicy ?	element	SHALL NOT be present when Layout/@Automated = "false" . Specifies automated fit policy for the page cell described by the MarkObject . When present, MarkObject/@TrimSize SHALL also be present in the MarkObject , and represents the cell size for this MarkObject .
IdentificationField *	element	Specific information about this kind of mark object. See below for information regarding dynamically generated marks.
JobField ?	element	JobField specifies the metadata of a given dynamic slug line.
MarkActivation *	element	Rules about when to apply the mark in an automated Layout . If no MarkActivation is specified, the MarkObject is unconditionally active. If multiple MarkActivation Elements are specified, all conditions SHALL be met for the mark to be active. MarkActivation SHALL NOT be specified unless Layout/@Automated = "true" .
RefAnchor ?	element	Details of the coordinate system that this mark is placed relative to. This MAY be either the sheet coordinate system or the coordinate system of a referenced MarkObject . If the anchor point in the referenced object (MarkObject or Sheet surface) is modified (e.g., due to a change in @TrimSize), the CTM of the placed object of this DeviceMark SHALL be modified accordingly. Note: RefAnchor does NOT modify the origin of the CTM of this MarkObject . It is only used to recalculate relative shifts.
RegisterMark *	element	Specific information about this kind of mark object. See below for information regarding dynamically generated marks.
ScavengerArea *	element	Specific information about this kind of mark object. See below for information regarding dynamically generated marks.

8.60.12 Element: CutMark

This element provides the means to position cut marks on the Sheet. After printing, these marks can be used to adapt the theoretical block positions (as specified in **CutBlock**) to the real position of the corresponding blocks on the printed Sheet.

Table 8-126: CutMark Resource (Sheet 1 of 2)

Name	Data Type	Description
Blocks ?	NMTOKENS	Values of the @BlockName Partition Attributes of the blocks defined by the CutMark Resource.

Table 8-126: CutMark Resource (Sheet 2 of 2)

Name	Data Type	Description
MarkType	enumeration	<p>Cut mark type.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>CrossCutMark</i> <i>TopVerticalCutMark</i> <i>BottomVerticalCutMark</i> <i>LeftHorizontalCutMark</i> <i>RightHorizontalCutMark</i> <i>LowerLeftCutMark</i> <i>UpperLeftCutMark</i> <i>LowerRightCutMark</i> <i>UpperRightCutMark</i>
Position	XYPair	Position of the logical center of the cut mark in the coordinates of the MarkObject that contains this mark. Note that the logical center of the cut mark does not always directly specify the center of the visible cut mark symbol.

Table 8-127: Cut mark types as specified by CutMark/@MarkType (Sheet 1 of 2)

Symbol	Mark type Value	Position of Symbol
	"CrossCutMark"	Centered at logical position
	"TopVerticalCutMark"	Slightly above logical position
	"BottomVerticalCutMark"	Slightly below logical position
	"LeftHorizontalCutMark"	Slightly to the left of logical position
	"RightHorizontalCutMark"	Slightly to the right of logical position
	"LowerLeftCutMark"	Corner at logical position
	"UpperLeftCutMark"	Corner at logical position
	"LowerRightCutMark"	Corner at logical position

Table 8-127: Cut mark types as specified by CutMark/@MarkType (Sheet 2 of 2)

Symbol	MarkType Value	Position of Symbol
	"UpperRightCutMark"	Corner at logical position

8.60.13 Element: FillMark

MarkActivation

Table 8-128: FillMark Element

Name	Data Type	Description
<i>KnockoutBleed</i> ?	double	Bleed in points that the fill should grow into (positive values) the knockout area. Note: this attribute implies the same bleed for all separations.
<i>KnockoutRefs</i> ?	IDREFS	Reference to the PlacedObject Elements that SHALL not be filled by this FillMark. The knockout boundaries are implied by the value of <i>@KnockoutSource</i> .
<i>KnockoutSource</i>	enumeration	Definition of the source of the knockout from the referenced PlacedObject Elements. Allowed values are: <i>ClipPath</i> – Use the Clip Path as defined by the referenced PlacedObject/ <i>@ClipPath</i> . <i>SourceClipPath</i> – Use the Clip Path as defined by the referenced PlacedObject/ <i>@SourceClipPath</i> . <i>TrimClipPath</i> – Use the Clip Path as defined by the referenced PlacedObject/ <i>@TrimClipPath</i> . <i>TrimBox</i> – Use the Clip Path as defined by the referenced PlacedObject/ <i>@TrimCTM</i> and PlacedObject/ <i>@TrimSize</i> .
<i>MarkColor</i> *	element	Definition of the separations used to fill the mark.

8.60.14 Element: MarkActivation

MarkActivation specifies condition when to apply the mark in an automated Layout.

Table 8-129: MarkActivation Element

Name	Data Type	Description
<i>Context</i>	NMTOKEN	<p>The context in which the iteration is counted.</p> <p>Values include:</p> <p><i>CollectSheetIndex</i> – a parameter maintained by the imposition engine to count sheets (e.g., in the context of a signature). Its value starts at 0 and is incremented by one for each sheet. If Layout/@MaxCollect is specified, its maximum value is one less than Layout/@MaxCollect. Otherwise, it continues to increment per sheet until completion of the page-pool/page-pool-list processing through the Imposition Template. See Section 6.4.18, “Imposition” on page 372.</p> <p><i>DocIndex</i> – a Partition Key.</p> <p><i>SetDocIndex</i> – a Partition Key.</p> <p><i>SetIndex</i> – a Partition Key.</p> <p><i>SheetIndex</i> – a Partition Key.</p> <p><i>SubDocIndex0, ...</i> – a parameter maintained by the imposition engine. See Section 6.4.18, “Imposition” on page 372.</p>
<i>Index</i>	IntegerRange	The enclosing MarkObject is active and its specified Mark SHALL be imaged if the value of the variable specified by @Context is equal to one of the values of this Attribute. Negative values in this list specify indexes that are counted from the back. For instance, if @Context = "SetDocIndex" and @Index = "-1" , the mark is active on the last Document of every Set.

8.60.14.1 Dynamic Marks

JobField and **@Ord** are mutually exclusive within one **MarkObject**.

The Elements marked as Dynamic marks in the table above can be used for three purposes:

- If **@Ord** is specified, the PDL of the mark is provided by the **RunList (Marks)** and the dynamic mark Subelements provide metadata about the mark to a press Controller or bindery equipment. This is the usual behavior of existing imposition engines. A single **MarkObject** SHALL NOT contain multiple mark Subelements that are represented by the same PDL, for instance there MAY be only one **Marks** layer for an entire surface.
- If **@Ord** is not present, but **JobField** is present, an Imposition Device SHOULD dynamically generate the mark based on information in **JobField**.
- If none of **@Ord** and **JobField** are present, a mark SHOULD be dynamically drawn based on the information within the Subelement. The marks are positioned relative to the **@CTM** of the **MarkObject**. A single **MarkObject** SHOULD NOT contain multiple dynamic mark Subelements. Note that the **XJDF** specification of dynamic marks other than **JobField** are in flux and that the behavior described here might change in future versions of **XJDF**.

8.60.15 Element: DynamicField

DynamicField provides a description of dynamic text replacements for a **MarkObject** Element. This Element is to be used for production purposes such as defining bar codes for variable data printing. **DynamicField** Elements are not intended as placeholders for actual content such as addresses. Rather, they are marks with dynamic data such as time stamps and database information. Dynamic objects are **MarkObject** Elements with additional OPTIONAL **DynamicField** Elements that define text replacement.

Table 8-130: DynamicField Element

Name	Data Type	Description
<i>Format</i>	string	Format string in C printf format that defines the replacement. Values may contain variables from: Appendix I, “Generating strings with Format and Template” on page 1203.
<i>ReplaceField?</i>	string	String that SHALL be replaced by the instantiated text expression as defined by the <i>@Format</i> and <i>@Template</i> Attributes in the file referenced by <i>MarkObject/@Ord</i> . If <i>@ReplaceField</i> is not specified, the Device that processes the <i>DynamicField</i> SHALL format the <i>DynamicField</i> .
<i>Template</i>	string	Comma separated list of variables to define a sequence of variables consumed by <i>@Format</i> . Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.

Example 8-14: Layout: DynamicField Element

In this example, the text “xxx” in the file MyReplace.pdf would be replaced by the sentence “Replacement Text for Joe and John go in here at 14:00 on Mar-31-2000”. MyReplace.pdf is placed at the position defined by the *@CTM* of the *MarkObject* and Variable.pdf is placed at the position defined by the *@CTM* of the *ContentObject*.

TBD 2.x Example.

```
<RunList Class="Parameter" ID="L3" PartIDKeys="Run" Status="Available">
  <MetadataMap DataType="string" Name="i1" ValueFormat="%s"
    ValueTemplate="s1">
    <!--This expression maps the value of /Dokument/Rezipient/@Name to a
        variable "s1"-->
    <Expr Name="s1" Path="/Dokument/Rezipient/@Name"/>
  </MetadataMap>
  <LayoutElement ElementType="Graphic">
    <FileSpec URL="File:///Variable.pdf"/>
  </LayoutElement>
</RunList>
<Layout Class="Parameter" ID="Link0003" Status="Available">
  <!--The MarkObject in the Layout hierarchy: -->
  <ContentObject CTM="1 0 0 1 0 0" Ord="0"/>
  <MarkObject CTM="1 0 0 1 10 10">
    <LayoutElement ElementType="Graphic">
      <FileSpec URL="File:///MyReplace.pdf"/>
    </LayoutElement>
    <DynamicField
      Format="Replacement Text for %s goes in here at %s on %s"
      Ord="0" ReplaceField="__xxx__" Template="i1,Time,Date"/>
    <DynamicField Format="More Replacement Text for %s go in here"
      Ord="0" ReplaceField="__yyy__" Template="SignatureName"/>
  </MarkObject>
</Layout>
```

8.60.16 More about Layout**8.60.16.0.1 Partition Key restrictions:**

If *MarkObject* and *ContentObject* are placed onto a **Layout**, at least **Resource/Part** element SHALL exist and **Resource/Part/@Side** SHALL be specified.

All *MarkObject* and *ContentObject* Elements that are intended to be imaged on one Surface SHALL be specified in one partition.

```

<Layout Class="Parameter" ID="L3" Status="Available"
  PartIDKeys="SignatureName SheetName Side">
  <!-- INVALID, this PlacedObject is not in a leaf partition and not used -->
  <!-- since it is overwritten by <MarkObject Ord="1"> -->
  <MarkObject Ord="0" CTM="0.0 1.0 -1.0 0.0 176.69 23.62" >
    <RegisterMark Center="0.0 0.0" MarkType="Cross" MarkUsage="PaperPath" />
  </MarkObject>
  <Layout SignatureName="Sig00">
    <!-- INVALID, this PlacedObject is not in a leaf partition -->
    <MarkObject Ord="1" CTM="0.0 1.0 -1.0 0.0 176.69 23.62" >
      <RegisterMark Center="0.0 0.0" MarkType="Cross"
        MarkUsage="PaperPath" />
    </MarkObject>
    <Layout SheetName="Sheet00">
      <Layout Side="Front">
        <MarkObject Ord="2" CTM="0.0 1.0 -1.0 0.0 176.69 23.62" >
          <RegisterMark Center="0.0 0.0" MarkType="Arc"
            MarkUsage="PaperPath" />
        </MarkObject>
        <ContentObject CTM="0.0 1.0 -1.0 0.0 176.69 23.62" Ord="0" />
      </Layout>
    </Layout>
    <Layout SheetName="Sheet01">
      <Layout Side="Front">
        <!-- Not clear whether this is layered over or under
            <MarkObject Ord="0">
          -->
        <ContentObject CTM="0.0 1.0 -1.0 0.0 176.69 23.62" Ord="0" />
      </Layout>
    </Layout>
  </Layout>
</Layout>

<Layout Class="Parameter" ID="L3" Status="Available"
  PartIDKeys="SignatureName SheetName Side">
  <Layout SignatureName="Sig00">
    <Layout SheetName="Sheet00">
      <Layout Side="Front">
        <MarkObject Ord="2" CTM="0.0 1.0 -1.0 0.0 176.69 23.62" >
          <RegisterMark Center="0.0 0.0" MarkType="Arc"
            MarkUsage="PaperPath"/>
        </MarkObject>
        <ContentObject CTM="0.0 1.0 -1.0 0.0 176.69 23.62" Ord="0" />
      </Layout>
    </Layout>
    <Layout SheetName="Sheet01">
      <Layout Side="Front">
        <MarkObject Ord="1" CTM="0.0 1.0 -1.0 0.0 176.69 23.62" >
          <RegisterMark Center="0.0 0.0" MarkType="Cross"
            MarkUsage="PaperPath" />
        </MarkObject>
        <ContentObject CTM="0.0 1.0 -1.0 0.0 176.69 23.62" Ord="0" />
      </Layout>
    </Layout>
  </Layout>
</Layout>

```

8.60.16.1 CTM Definitions

The following are explanations of the terms used in this section and beyond:

- **Dimensions of object** – The width and height of either the box defined to include all drawings for this file format, or the artificial box that includes these drawings for file formats that have no clearly defined box for this.
- **Trim box of the Signature page** – A rectangle that indicates where the trim box of object is to be positioned. This is the equivalent to the area the user is intended to see in the final product. Positioning the trim box of the object inside the trim box of the Signature page is implementation-specific (usually it is centered).
- **Trim box of the object** – A rectangle that is PDL-specific that indicates the area of the object that indicates the intended trimming area.

8.60.16.2 Finding the Trim Box of an Object

The **RunList/@SourceTrimBox** always takes precedence over boxes defined inside the file. Make sure that **RunList/@SourceTrimBox** is updated after replacing Elements. The following is a list of names used for the real trim box in various file formats:

- PostScript (PS) – **PageSize**
- Encapsulated PostScript (EPS) – **CropBox**
- Portable Document Format (PDF) – **TrimBox**
- Raster files – entire area

If this information is not available, alternative sources for trim box information can include (but these boxes might not be correct in all cases):

- EPS – **HiResBoundingBox** then **BoundingBox**
- PDF – **CropBox** then **MediaBox**

8.60.16.3 Using Ord to Reference Elements in RunList Resources

The **@Ord** Attribute in **ContentObject** or **MarkObject** Elements represents a reference to a *logical* element in a **RunList**. The index is incremented for every page of the **RunList** with **@IsPage = "true"**. The reference is not changed by repartitioning the **RunList**. The content and marks **RunList** are referenced independently. The following examples illustrate the usage of **@Ord**.

Example 8-15: RunList: Simple Multi-File Unseparated RunList

This example specifies all pages contained in File1.pdf and File2.pdf. File 1 has 6 pages, file 2 has an unknown number of pages.

TBD 2.x Example.

```
<RunList Class="Parameter" ID="L3" PartIDKeys="Run" Status="Available">
  <RunList NPage="6" Pages="0 ~ 5" Run="1">
    <LayoutElement>
      <FileSpec URL="File:///File1.pdf"/>
    </LayoutElement>
  </RunList>
  <RunList Pages="0 ~ -1" Run="2">
    <LayoutElement>
      <FileSpec URL="File:///File2.pdf"/>
    </LayoutElement>
  </RunList>
</RunList>
```

Table 8-131: Example (1) of Ord Attribute in PlacedObject Elements

Ord	File	Page	Ord	File	Page
0	File1	0	1	File1	1
2	File1	2	3	File1	3
4	File1	4	5	File1	5
6	File2	0	7	File2	1
8	File2	2	(n)	File2	(n - 6)

Example 8-16: RunList: Simple Multi-File Separated RunList

This example specifies two pages contained in Presep.pdf and following that, pages 1, 3 and 5 of each preseparated file.

TBD 2.x Example.

```
<RunList Class="Parameter" ID="Link0003" PartIDKeys="Run Separation"
         Status="Available">
  <RunList NPage="2" Run="1" SkipPage="3">
    <LayoutElement>
      <FileSpec URL="File:///Presep.pdf"/>
    </LayoutElement>
    <RunList FirstPage="0" IsPage="false" Separation="Cyan"/>
    <RunList FirstPage="1" IsPage="false" Separation="Magenta"/>
    <RunList FirstPage="2" IsPage="false" Separation="Yellow"/>
    <RunList FirstPage="3" IsPage="false" Separation="Black"/>
  </RunList>
  <RunList IsPage="true" Pages="1 3 5" Run="2">
    <RunList IsPage="false" Separation="Cyan">
      <LayoutElement>
        <FileSpec URL="File:///Cyan2.pdf"/>
      </LayoutElement>
    </RunList>
    <RunList IsPage="false" Separation="Magenta">
      <LayoutElement>
        <FileSpec URL="File:///Magenta2.pdf"/>
      </LayoutElement>
    </RunList>
    <RunList IsPage="false" Separation="Yellow">
      <LayoutElement>
        <FileSpec URL="File:///Yellow2.pdf"/>
      </LayoutElement>
    </RunList>
    <RunList IsPage="false" Separation="Black">
      <LayoutElement>
        <FileSpec URL="File:///Black2.pdf"/>
      </LayoutElement>
    </RunList>
  </RunList>
</RunList>
```

Table 8-132: Example (2) of Ord Attribute in PlacedObject Elements (Sheet 1 of 2)

Ord	File	Page	Separation	Ord	File	Page	Separation
0	PreSep	0	Cyan	0	Presep	1	Magenta

Table 8-132: Example (2) of Ord Attribute in PlacedObject Elements (Sheet 2 of 2)

Ord	File	Page	Separation	Ord	File	Page	Separation
0	PreSep	2	Yellow	0	Presep	3	Black
1	PreSep	4	Cyan	1	Presep	5	Magenta
1	PreSep	6	Yellow	1	Presep	7	Black
2	Cyan2	1	Cyan	2	Magenta2	1	Magenta
2	Yellow2	1	Yellow	2	Black2	1	Black
3	Cyan2	3	Cyan	3	Magenta2	3	Magenta
3	Yellow2	3	Yellow	3	Black2	3	Black
4	Cyan2	5	Cyan	4	Magenta2	5	Magenta
4	Yellow2	5	Yellow	4	Black2	5	Black

8.60.16.4 Using Expressions in the OrdExpression Attribute

Expressions can use the operators $+$, $-$, $*$, $/$, $\%$ and parentheses, operating on integers and two variables: s for Signature number (starting at 0) and n for number of pages to be imposed in one document. Signature number denotes the number of times that a complete set of placed objects has been filled with content from the run list. The operators have the same meaning as in the C programming language. Expressions are evaluated with normal “C” operator precedence. Multiplication SHALL be expressed by explicitly including the * operator (i.e., use “ $2*s$ ”, not “ $2 s$ ”). Remainders are discarded.

Example 8-17: OrdExpression

Saddle stitched booklet for variable page length documents.

The following describes the OrdExpressions for a booklet with varying page lengths. The example page assignments are for a book of 13-16 pages.

TBD 2.x Example.

Front:

```
OrdExpression = 2*s      0  2  4  6
OrdExpression = 4*((n+3)/4) (s*2)-11513119
```

Back:

```
OrdExpression = 2*s+1   1  3  5  7
OrdExpression = 4*((n+3)/4) (s*2)-21412108
```

Example 8-18: DocOrd Usage

Two-sided business cards 4/Sheet

The following describes the Ord + DocOrd usage for a 4-up step + repeat business card

TBD 2.x Example.

```
MaxDocOrd = 4
```

Front:

```
Ord = 0 DocOrd = 0
Ord = 0 DocOrd = 1
Ord = 0 DocOrd = 2
Ord = 0 DocOrd = 3
```

Back:

```
Ord = 1 DocOrd = 0
Ord = 1 DocOrd = 1
Ord = 1 DocOrd = 2
Ord = 1 DocOrd = 3
```

8.61 LayoutElementProductionParams

This Resource is needed for *LayoutElementProduction*. This contains detailed information about the type of *RunList* to be produced.

Resource Properties

Resource referenced by:

Input of Processes: *LayoutElementProduction*

Output of Processes:

Table 8-133: LayoutElementProductionParams Resource

Name	Data Type	Description
<i>ContentRef?</i>	IDREF	Description of the specific parameters for generating a <i>RunList</i> .
<i>ShapeDefRef?</i>	IDREF	A Resource describing the shape of the <i>RunList</i> to be produced.
<i>FileSpec</i> (DataList) ?	element	References a data list containing record information for variable data production. The format of the referenced data is implementation specific.
<i>FileSpec</i> (PreflightProfile) ?	element	External preflight profile that describes the rules that the completed <i>RunList</i> SHALL adhere to.

Example 8-19: LayoutElementProductionParams: Page Shape

TBD 2.x Example.

```
<!-- Page Shape Sample
    Date: Aug 2, 2007 Version: 2
    A page with a certain size -->
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n001"
    Type="LayoutElementProduction"
    Status="Waiting" DescriptiveName="Page sample for shape"
    JobPartID="ID34" Version="1.4">
    <ResourcePool>
        <LayoutElementProductionParams Class="Parameter" ID="LEPParams"
            Status="Available" />
        <LayoutElement Class="Parameter" ID="LayElOut" Status="Unavailable"
            SourceMediaBox="0 0 595.27 822.05"
            SourceTrimBox="28.34 28.34 566.93 793.71"/>
    </ResourcePool>
    <ResourceLinkPool>
        <LayoutElementProductionParamsLink rRef="LEPParams" Usage="Input"/>
        <LayoutElementLink rRef="LayElOut" Usage="Output"/>
    </ResourceLinkPool>
    <AuditPool>
        <Created AgentName="XYZ Corporation"TimeStamp="2006-01-09T09:00:00+01:00"/>
    </AuditPool>
</JDF>
```

Example 8-20: LayoutElementProductionParams: Label Shape

TBD 2.x Example.

```
<!-- Shape Sample for a label with a cut line
    Date: Jan 9, 2005 Version: 1.00
    A page with a certain size -->
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n001"
    Type="LayoutElementProduction"
    Status="Waiting" DescriptiveName="Page sample for shape"
```

```



```

Example 8-21: LayoutElementProductionParams: Box Shape

TBD 2.x Example.

```

<!-- Shape Sample for a box defined by a CAD file
     Date: Jan 9, 2005 Version: 1.00
     A page with a certain size -->
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n001"
    Type="LayoutElementProduction"
    Status="Waiting" JobPartID="ID100"
    DescriptiveName="Page sample for shape" Version="1.4">
<ResourcePool>
    <LayoutElementProductionParams Class="Parameter" ID="LEPParams"
        Status="Available">
        <ShapeDef>
            <FileSpec URL="file:///myserver/myshare/olive.dd3"/>
        </ShapeDef>
    </LayoutElementProductionParams>
    <LayoutElement Class="Parameter" ID="LayElOut" Status="Unavailable"/>
</ResourcePool>
<ResourceLinkPool>
    <LayoutElementProductionParamsLink rRef="LEPParams" Usage="Input"/>
    <LayoutElementLink rRef="LayElOut" Usage="Output"/>
</ResourceLinkPool>
<AuditPool>
    <Created AgentName="ZYX Corporation" TimeStamp="2006-01-09T09:00:00+01:00"/>
</AuditPool>
</JDF>

```

TBD 2.x Example.

```

<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n000002"
    JobPartID="n000002" Status="Waiting" Type="LayoutElementProduction"
    Version="1.4"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="LayoutElementProduction">

```

```

<!--Generated by the CIP4 Java open source JDF Library version :  

    CIP4 JDF Writer Java 1.3 BLD 46-->  

<AuditPool>  

    <Created AgentName="CIP4 JDF Writer Java" AgentVersion="1.3 BLD 46"  

        ID="a000003"  

        TimeStamp="2007-09-05T18:20:31+02:00"/>  

</AuditPool>  

<ResourcePool>  

    <RunList Class="Parameter" ID="r000004" Status="Unavailable">  

        <LayoutElement Class="Parameter">  

            <FileSpec Class="Parameter" MimeType="application/pdf" URL="output.pdf"/>  

        </LayoutElement>  

    </RunList>  

    <LayoutElementProductionParams Class="Parameter" ID="r000005"  

        Status="Unavailable">  

        <!--This is a "well placed" CTM defined mark  

            The anchor defines the 0,0 point to be transformed  

            The element to be placed is referenced by LayoutElement/FileSpec/URL  

        -->  

        <LayoutElementPart>  

            <PositionObj Anchor="BottomLeft" CTM="1 0 0 1 0 0" PageRange="0"  

                PositionPolicy="Exact">  

                <RefAnchor Anchor="BottomLeft" AnchorType="Parent"/>  

            </PositionObj>  

            <LayoutElement Class="Parameter">  

                <FileSpec Class="Parameter" MimeType="application/pdf"  

                    URL="bkg.pdf"/>  

            </LayoutElement>  

        </LayoutElementPart>  

        <!--This is a "roughly placed" reservation in the middle of the page-->  

        <LayoutElementPart ID="1000006">  

            <PositionObj Anchor="Center" PageRange="0" PositionPolicy="Free">  

                <RefAnchor Anchor="Center" AnchorType="Parent"/>  

            </PositionObj>  

            <LayoutElement Class="Parameter" ElementType="Image">  

                <Comment ID="c000007">  

                    Please add an image of a palm tree on a beach here!  

                </Comment>  

            </LayoutElement>  

        </LayoutElementPart>  

        <!--This is a "roughly placed" reservation 36 points below the previous  

            image; NextPosition points from Anchor on this to NextAnchor on next,  

            i.e. a positive vector specifies that next is shifted in the positive  

            direction in the parent (in this case page) coordinate system  

        -->  

        <LayoutElementPart>  

            <PositionObj Anchor="TopCenter" CTM="1 0 0 1 0 36"  

                PageRange="0" PositionPolicy="Free">  

                <RefAnchor Anchor="BottomCenter" AnchorType="Sibling"  

                    rRef="1000006"/>  

            </PositionObj>  

            <LayoutElement Class="Parameter" ElementType="Image">  

                <Comment ID="c000008">  

                    Please add an image of a beach ball below the palm tree!  

                </Comment>  

            </LayoutElement>  

        </LayoutElementPart>  

        <!--This is a "well placed" CTM defined mark. The anchor defines the
    
```

```

    0,0 point used as the RefAnchor for the element to be transformed
-->
<LayoutElementPart>
    <PositionObj Anchor="BottomLeft" CTM="1 0 0 1 2 3" PageRange="0"
        PositionPolicy="Exact">
        <RefAnchor Anchor="BottomLeft" AnchorType="Parent"/>
    </PositionObj>
    <BarcodeProductionParams>
        <!--barcode details here-->
        <IdentificationField Encoding="Barcode" EncodingDetails="CODABAR" />
    </BarcodeProductionParams>
</LayoutElementPart>
<LayoutElementPart>
    <PositionObj Anchor="TopRight" PageRange="0" PositionPolicy="Exact">
        <RefAnchor Anchor="TopRight" AnchorType="Parent"/>
        <!--This is a "roughly placed" mark.
            The anchor at top right is placed at the right (=1.0) top(=1.0)
            position of the page. No rotation is specified
        -->
    </PositionObj>
    <BarcodeProductionParams>
        <!--barcode details here-->
        <IdentificationField Encoding="Barcode" EncodingDetails="CODABAR" />
    </BarcodeProductionParams>
</LayoutElementPart>
<!--This is a "roughly placed" container for marks
    The anchor at top left is defined in the !Unrotated! orientation.
    It is placed at the left (=0.0) bottom(=0.0) position of the page.
    The text flows bottom to top (=Rotate 90 = counterclockwise)
    do we need margins?
-->
<LayoutElementPart ID="1000009">
    <PositionObj Anchor="TopLeft" CTM="0 1 -1 0 0 0"
        PageRange="1" PositionPolicy="Free">
        <RefAnchor Anchor="BottomCenter" AnchorType="Parent"/>
    </PositionObj>
</LayoutElementPart>
<!--This is a barcode inside the previous container
    The anchor at bottom left is defined in the !Unrotated! orientation.
    It is placed at the left (=0.0) bottom(=0.0) position of the container.
-->
<LayoutElementPart ID="1000010">
    <PositionObj Anchor="BottomLeft" CTM="1 0 0 1 0 0">
        <RefAnchor Anchor="BottomLeft" AnchorType="Parent" rRef="1000009"/>
    </PositionObj>
    <BarcodeProductionParams>
        <!--barcode details here-->
        <IdentificationField Encoding="Barcode" EncodingDetails="CODABAR" />
    </BarcodeProductionParams>
</LayoutElementPart>
<!--This is a disclaimer text inside the previous container
    The anchor at top left is defined in the !Unrotated! orientation.
    The barcode and text are justified with their top margins and spaced
    by 72 points which corresponds to the left of the page because the
    container is rotated 90° AbsoluteSize specifies the size of the
    object in points
-->
<LayoutElementPart>

```

```

<PositionObj Size="300 200" Anchor="TopLeft" CTM="1 0 0 1 -72 0">
    <RefAnchor Anchor="TopRight" AnchorType="Sibling" rRef="1000010"/>
</PositionObj>
<LayoutElement Class="Parameter" ElementType="Text">
    <FileSpec Class="Parameter"
        URL="file:///myServer/disclaimers/de/aspirin.txt"/>
    </LayoutElement>
</LayoutElementPart>
<!--This is a "VERY roughly placed" piece of text somewhere on pages 2-3
    RelativeSize specifies the size of the object as a ratio of the size
    of the container
-->
<LayoutElementPart>
    <PositionObj PageRange="1 ~ 2" RelativeSize="0.8 0.5"/>
    <LayoutElement Class="Parameter" ElementType="Text">
        <Comment ID="c000011" Name="Instructions">
            Please add some text about
            the image of a palm tree on a beach here!
        </Comment>
    </LayoutElement>
</LayoutElementPart>
<!--This is another "VERY roughly placed" piece of text somewhere on
    pages 2-3; the text source is the JDF-->
<LayoutElementPart>
    <PositionObj PageRange="1 ~ 2"/>
    <LayoutElement Class="Parameter" ElementType="Text">
        <Comment ID="c000012" Name="TextInput">
            Laurum Ipsum Blah blah blah!
            btw. this is unformatted plain text and nothing else!
        </Comment>
    </LayoutElement>
</LayoutElementPart>
</LayoutElementProductionParams>
</ResourcePool>
<ResourceLinkPool>
    <RunListLink Usage="Output" rRef="r000004"/>
    <LayoutElementProductionParamsLink Usage="Input" rRef="r000005"/>
</ResourceLinkPool>
</JDF>

```

TBD 2.x Example.

```

<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="n000002"
    JobPartID="n000002" Status="Completed" Type="LayoutElementProduction"
    Version="1.4" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="LayoutElementProduction">
    <!--Generated by the CIP4 Java open source JDF Library version : CIP4 JDF
        Writer Java 1.3 BLD 47-->
    <AuditPool>
        <Created AgentName="CIP4 JDF Writer Java" AgentVersion="1.3 BLD 47"
            ID="a000003"
           TimeStamp="2007-10-11T20:23:18+02:00"/>
        <PhaseTime AgentName="CIP4 JDF Writer Java"
            AgentVersion="1.3 BLD 47"
            End="2007-10-11T20:23:23+02:00" ID="a000020"
            Start="2007-10-11T20:23:21+02:00" Status="InProgress"
            StatusDetails="Creative Work"TimeStamp="2007-10-11T20:23:21+02:00"/>
        <ProcessRun AgentName="CIP4 JDF Writer Java"
            AgentVersion="1.3 BLD 47"

```

```

    Duration="PT2S" End="2007-10-11T20:23:23+02:00"
    EndStatus="Completed" ID="a000024"
    Start="2007-10-11T20:23:21+02:00"TimeStamp="2007-10-11T20:23:23+02:00"/>
</AuditPool>
<ResourcePool>
    <RunList Class="Parameter" ID="r000004" Status="Unavailable">
        <LayoutElement Class="Parameter">
            <FileSpec Class="Parameter" MimeType="application/pdf" URL="output.pdf"/>
        </LayoutElement>
    </RunList>
    <LayoutElementProductionParams Class="Parameter" ID="r000005" Status="Unavailable">
        <Comment ID="c000006" Name="Instruction">
            Add any human readable instructions here
        </Comment>
        <ActionPool>
            <Action DescriptiveName="set number of pages to 4" ID="A000007" Severity="Error" TestRef="T000008"/>
            <Action DescriptiveName="set number of separations to 6 on page 0 and 3" ID="A000009" Severity="Error" TestRef="T000010">
                <PreflightAction SetRef="T000011"/>
            </Action>
            <Action DescriptiveName="separation to black only on page 1 and 2" ID="A000012" Severity="Error" TestRef="T000013">
                <PreflightAction SetRef="T000014"/>
            </Action>
            <Action DescriptiveName="set TrimBox to 8.5*11 Method 2" ID="A000015" Severity="Error" TestRef="T000016">
                <PreflightAction SetRef="T000017"/>
            </Action>
            <Action DescriptiveName="Warn when effective resolution<300 dpi" ID="A000018" Severity="Warning" TestRef="T000019"/>
        </ActionPool>
    <TestPool>
        <Test ID="T000008">
            <not>
                <IntegerEvaluation ValueList="4">
                    <BasicPreflightTest Name="NumberOfPages"/>
                </IntegerEvaluation>
            </not>
        </Test>
        <Test ID="T000010">
            <not>
                <StringEvaluation>
                    <BasicPreflightTest ListType="UniqueList" MaxOccurs="6" MinOccurs="6" Name="SeparationList"/>
                </StringEvaluation>
            </not>
        </Test>
        <Test ID="T000011">
            <IntegerEvaluation ValueList="0 3">
                <BasicPreflightTest Name="PageNumber"/>
            </IntegerEvaluation>
        </Test>
        <Test ID="T000013">

```

```

<not>
  <StringEvaluation>
    <BasicPreflightTest Name="SeparationList"/>
    <Value Value="Black"/>
  </StringEvaluation>
</not>
</Test>
<Test ID="T000014">
  <IntegerEvaluation ValueList="1 ~ 2">
    <BasicPreflightTest Name="PageNumber"/>
  </IntegerEvaluation>
</Test>
<Test ID="T000016">
  <not>
    <RectangleEvaluation ValueList="0 0 612 792">
      <BasicPreflightTest Name="PageBoxSize"/>
    </RectangleEvaluation>
  </not>
</Test>
<Test ID="T000017">
  <EnumerationEvaluation ValueList="TrimBox">
    <BasicPreflightTest Name="PageBoxName"/>
  </EnumerationEvaluation>
</Test>
<Test ID="T000019">
  <XYPairEvaluation ValueList="0 0 ~ 300 300">
    <BasicPreflightTest Name="EffectiveResolution"/>
  </XYPairEvaluation>
</Test>
</TestPool>
</LayoutElementProductionParams>
</ResourcePool>
<ResourceLinkPool>
  <RunListLink Usage="Output" rRef="r000004"/>
  <LayoutElementProductionParamsLink Usage="Input" rRef="r000005"/>
</ResourceLinkPool>
</JDF>

```

— Attribute: PageDistributionScheme

<LayoutPreparationParams Status="Available" Class="Parameter" ID="LPP_2"

Table 8-134: PageDistributionScheme Attribute Values (Sheet 1 of 2)

Value	Description
<i>Perfect</i>	Distribute finished pages onto a sequence of one or more Signatures in proper order for perfect binding. For this page distribution scheme, creep is usually not used.
<i>PerfectFront</i>	Distribute finished pages onto a sequence of one or more Signatures in proper order for perfect binding where only the reader order front pages respective of the finished product are placed in the signature layout. For left hand binding, only right facing page cells contain pages and left facing page cells are empty. For right hand binding, only left facing page cells contain pages and right facing page cells are empty. For top binding, only bottom facing page cells contain pages and top facing page cells are empty. For bottom binding, only top facing page cells contain pages and bottom facing page cells are empty. For this page distribution scheme, creep is usually not used.

Table 8-134: PageDistributionScheme Attribute Values (Sheet 2 of 2)

Value	Description
Saddle	Distribute finished pages onto a sequence of one or more imposition layouts in proper order for saddle stitch binding. For this page distribution scheme, creep is to be applied only to odd-numbered vertical gutters where any even-numbered gutters is to automatically creep in the opposite direction.
SaddleFront	Distribute finished pages onto a sequence of one or more imposition layouts in proper order for saddle stitch binding where only the reader order front pages respective of the finished product are placed in the signature layout. For left hand binding, only right facing page cells contain pages and left facing page cells are empty. For right hand binding, only left facing page cells contain pages and right facing page cells are empty. For top binding, only bottom facing page cells contain pages and top facing page cells are empty. For bottom binding, only top facing page cells contain pages and bottom facing page cells are empty. For this page distribution scheme, creep is to be applied only to odd-numbered vertical gutters where any even-numbered gutters is to automatically creep in the opposite direction.
Sequential	The finished pages are distributed onto the multi-up layout according to the value of the <i>@PresentationDirection</i> Attribute.

```

NumberUp="2 2" PageDistributionScheme="Saddle" FoldCatalog="F8-7"
FoldCatalogOrientation="Flip270" Sides="TwoSidedFlipY"
StepRepeat="1 1 1" SurfaceContentsBox="0 0 612 792" BindingEdge="Left"
VerticalCreep="0" GutterMinimumLimit="5 5" CreepValue="0 -5"
Gutter="20 20" FinishingOrder="FoldCollect" FrontMarkList="CutMark">
<!-- Note: the value of some attributes in LayoutPreparationParams and
subElements relate to symbols in the above Figure:
SurfaceContentsBox="SCBx0 SCBy0 SCBx1 SCBy1"
GutterMinimumLimit="hml vml"
CreepValue="0 -vc"
Gutter="hg vg"
TrimSize="m n"
ShiftFront="SFx SFy"

-->
<PageCell TrimSize="612 792">
    <ImageShift PositionX="Spine" PositionY="Center" />
</PageCell>
<ImageShift PositionY="Bottom" PositionX="Left" ShiftFront="20 20"/>
</LayoutPreparationParams>
<LayoutPreparationParams Class="Parameter" ID="LPP_1" Status="Available"
    NumberUp="4 1" PageDistributionScheme="Saddle" FoldCatalog="F4-1"
    FoldCatalogOrientation="Flip0" Sides="TwoSidedFlipY" StepRepeat="2 1 1"
    SurfaceContentsBox="0 0 612 792" VerticalCreep="0 2"
    ImplicitGutter="0 30" ImplicitGutterMinimumLimit="0 20" CreepValue="0 5"
    Gutter="0 10" FinishingOrder="GatherFold" FrontMarkList="CutMark">
    <!--Note: folding pattern F4-1 applies to each of the two 2x1
signatures
Note: step and repeat by two in X direction logically divides grid
into two 2x1 signatures
Note: first (VC0) and third (VC2) vertical gutters are explicitly
creeping and the rest (~VC) are implicitly creeping
Note: Positive vertical creep value indicates initial gutter
Widths of inner most Sheet
Note: cut marks are located relative to largest page cell grid
trim box
Note: the value of some attributes in LayoutPreparationParams and
subElements relate to symbols in the above Figure:

```

```

    SurfaceContentsBox="SCBx0 SCBx1 SCBy0 SCBy1"
    ImplicitGutter="0 vig"
    ImplicitGutterMinimumLimit="0 vigl"
    CreepValue="0 +vc"
    Gutter="0 vg"
    TrimSize="m n"
    ShiftFront="SFx SFy"
  -->
<PageCell TrimSize="612 792">
  <ImageShift PositionX="Spine" PositionY="Bottom"/>
</PageCell>
<ImageShift PositionY="Bottom" PositionX="Left" ShiftFront="20 20"/>
</LayoutPreparationParams>

```

8.62 LayoutShift

Resource Properties

Resource referenced by:

—

Input of Processes:

LayoutShifting

Output of Processes:

—

Table 8-135: LayoutShift Resource

Name	Data Type	Description
ShiftPoint +	element	Description of separation dependent transformations for a given point on the Layout .

8.62.1 Element: ShiftPoint

Table 8-136: ShiftPoint Element

Name	Data Type	Description
CTM	matrix	@CTM that SHALL be applied to the Separation after all other transformations.
Position	XYPair	Point that this ShiftPoint applies to. Note: the interpolation algorithm between ShiftPoint positions is implementation dependent.

Example 8-22: LayoutShift

Example of absolute positions with @Position

TBD 2.x Example.

```

<!--LayoutShift SHOULD be partitioned: at least Side and Separation
   will make sense -->
<LayoutShift ID="r000005" Class="Parameter" Status="Unavailable"
  PartIDKeys="Side Separation" >
  <!--LayoutShift SHOULD be partitioned: at least Side and Separation
      will make sense-->
  <!--Note that the interpolation algorithm between positions is
      implementation dependent-->
  <LayoutShift Side="Front">
    <LayoutShift Separation="Cyan">
      <ShiftPoint CTM="1 0 0 1 0 0" Position="360 500"/>
      <ShiftPoint CTM="1 0 0 1 0 2" Position="1800 500"/>
      <ShiftPoint CTM="1 0 0 1 1 0" Position="360 1500"/>

```

```

<ShiftPoint CTM="1 0 0 1 1 2" Position="1800 1500"/>
<ShiftPoint CTM="1 0 0 1 2 0" Position="360 2500"/>
<ShiftPoint CTM="1 0 0 1 2 2" Position="1800 2500"/>
<ShiftPoint CTM="1 0 0 1 3 0" Position="360 3500"/>
<ShiftPoint CTM="1 0 0 1 3 2" Position="1800 3500"/>
</LayoutShift>
<LayoutShift Separation="Magenta">
    <ShiftPoint CTM="1 0 0 1 1 1" Position="360 500"/>
    <ShiftPoint CTM="1 0 0 1 1 3" Position="1800 500"/>
    <ShiftPoint CTM="1 0 0 1 2 1" Position="360 1500"/>
    <ShiftPoint CTM="1 0 0 1 2 3" Position="1800 1500"/>
    <ShiftPoint CTM="1 0 0 1 3 1" Position="360 2500"/>
    <ShiftPoint CTM="1 0 0 1 3 3" Position="1800 2500"/>
    <ShiftPoint CTM="1 0 0 1 4 1" Position="360 3500"/>
    <ShiftPoint CTM="1 0 0 1 4 3" Position="1800 3500"/>
</LayoutShift>
<LayoutShift Separation="Yellow">
    <ShiftPoint CTM="1 0 0 1 2 2" Position="360 500"/>
    <ShiftPoint CTM="1 0 0 1 2 4" Position="1800 500"/>
    <ShiftPoint CTM="1 0 0 1 3 2" Position="360 1500"/>
    <ShiftPoint CTM="1 0 0 1 3 4" Position="1800 1500"/>
    <ShiftPoint CTM="1 0 0 1 4 2" Position="360 2500"/>
    <ShiftPoint CTM="1 0 0 1 4 4" Position="1800 2500"/>
    <ShiftPoint CTM="1 0 0 1 5 2" Position="360 3500"/>
    <ShiftPoint CTM="1 0 0 1 5 4" Position="1800 3500"/>
</LayoutShift>
<LayoutShift Separation="Black">
    <ShiftPoint CTM="1 0 0 1 3 3" Position="360 500"/>
    <ShiftPoint CTM="1 0 0 1 3 5" Position="1800 500"/>
    <ShiftPoint CTM="1 0 0 1 4 3" Position="360 1500"/>
    <ShiftPoint CTM="1 0 0 1 4 5" Position="1800 1500"/>
    <ShiftPoint CTM="1 0 0 1 5 3" Position="360 2500"/>
    <ShiftPoint CTM="1 0 0 1 5 5" Position="1800 2500"/>
    <ShiftPoint CTM="1 0 0 1 6 3" Position="360 3500"/>
    <ShiftPoint CTM="1 0 0 1 6 5" Position="1800 3500"/>
</LayoutShift>
</LayoutShift>
</LayoutShift>

```

8.63 LooseBindingParams

This Parameter describes the details of the ***LooseBinding*** Process.

Resource Properties

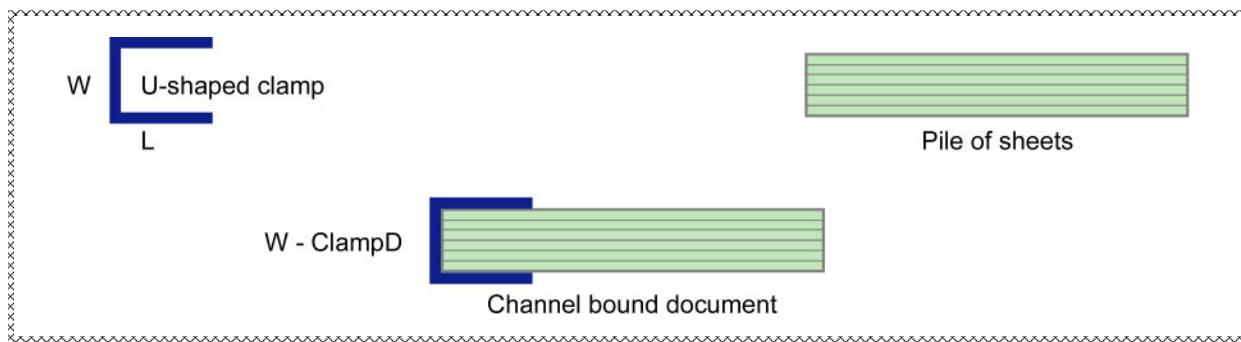
Resource referenced by: —

Intent Pairing: **BindingIntent**

Input of Processes: ***LooseBinding***

Output of Processes: —

Figure 8-46 depicts the ChannelBinding process.

Figure 8-37: Parameters used for channel binding

The symbols W, L and ClampD of Figure 8-46 are described by the Attributes *@ClampD* and *@ClampSize* of the table below.

Table 8-137: LooseBindingParams Resource (Sheet 1 of 3)

Name	Data Type	Description
<i>BinderBrand</i> ?	string	Specifies the name of the binder: ChannelBinding: the clamp (or preassembled cover with clamp) manufacturer and the specific item. CoilBinding: the coil manufacturer and the specific item. PlasticCombBinding: the comb manufacturer and the specific item. RingBinding: binder manufacturer and the specific item. Note: for 2.x, <i>@Brand</i> replaces <i>@BinderName</i> in 1.x. StripBinding: strip manufacturer and the specific item. WireCombBinding: the comb manufacturer (e.g., <i>Wire-O</i> ®) and the specific item.
<i>BindingType</i> ?	NMTOKEN	Type of binding that is performed. Values include: <i>ChannelBinding</i> – Metal clamps are used to bind sheets. <i>CoilBinding</i> – Metal wire, wire with plastic or pure plastic is used to fasten pre-punched sheets of paper, cardboard or other materials. <i>CombBinding</i> – Plastic insert wraps through pre-punched holes in the substrate. <i>RingBinding</i> – Pre-punched sheets are placed in a ring binder. <i>StripBinding</i> – Hard plastic strips are held together by plastic pins, which in turn are bound to the strips with heat.
<i>CoverColor</i> ?	NamedColor	Color of the cover used for LooseBinding . See also <i>@SpineColor</i> .
<i>CoverColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color of the cover used for LooseBinding . If <i>@CoverColorDetails</i> is supplied, <i>@CoverColor</i> SHOULD also be supplied.

Table 8-137: LooseBindingParams Resource (Sheet 2 of 3)

Name	Data Type	Description
<i>CoverMaterial</i> ?	NMTOKEN	<p>The following describe binder materials used.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Cardboard</i> – Cardboard with no covering. <i>ClothCovered</i> – Cardboard with cloth covering. <i>Plastic</i> – Binder cover fabricated from solid plastic Sheet material (e.g., PVC Sheet). Note: for 1.x, called "<i>PVC</i>". <i>VinylCovered</i> – Cardboard with colored vinyl covering. Note: for 1.x, called "<i>PVCCovered</i>".
<i>FlipBackCover</i> ?	boolean	<p>The spine is typically hidden between the last page of the Component and the back cover. Flip the back cover after the wire was “closed” or keep it open. The latter makes sense if further processing is needed (e.g., inserting a CD) before closing the book.</p> <p>Note: @<i>FlipBackCover</i> is for “<i>WireCombBinding</i>” only.</p>
<i>Length</i> ?	double	<p>The length of the pin is determined by the height of the pile of Sheets to be bound.</p> <p>Note: @<i>Length</i> is for “<i>StripBinding</i>” only.</p>
<i>Material</i> ?	enumeration	<p>The material used for forming the coil binding with “<i>CoilBinding</i>” or wire comb binding with “<i>WireCombBinding</i>”.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Steel</i> – Plain steel. For both binding types. <i>ColorCoatedSteel</i> – Coated steel. For both binding types. <i>Plastic</i> – any kind of plastic. For “<i>CoilBinding</i>” only <p>Note: “<i>LaqueredSteel</i>”, “<i>TinnedSteel</i>”, “<i>ZincsSteel</i>”, for 1.x “<i>CoilBinding</i>” and “<i>WireCombBinding</i>” renamed to “<i>ColorCoatedSteel</i>”.</p> <p>Note: “<i>NylonCoatedSteel</i>” for 1.x “<i>CoilBinding</i>” renamed to “<i>ColorCoatedSteel</i>”.</p> <p>Note: “<i>PVC</i>” for 1.x “<i>CoilBinding</i>” renamed to “<i>Plastic</i>”.</p>
<i>SpineColor</i> ?	NamedColor	<p>Determines the color of the respective Binder.: ChannelBinding: the clamp/cover. If @<i>ClampSystem</i> = “true”, then the color of the cover is also meant.</p> <p>CoilBinding: the coil.</p> <p>PlasticCombBinding: the plastic comb.</p> <p>RingBinding: the binders spine; see also @<i>BinderColor</i>.</p> <p>StripBinding: the strip.</p> <p>WireCombBinding: the wire comb.</p>
<i>SpineColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color of the spine. If @ <i>SpineColorDetails</i> is supplied, @ <i>SpineColor</i> SHOULD also be supplied.

Table 8-137: LooseBindingParams Resource (Sheet 3 of 3)

Name	Data Type	Description
HolePattern *	element	Details of the holes for binding. Values for HolePattern/@ <i>Pattern</i> SHOULD be taken from Appendix M, “JDF XJDF/CIP4 Hole Pattern Catalog” on page 1271 and SHOULD follow the conventions specified in Table M-1, “Naming Scheme for Hole Patterns”
ChannelBindingDetails ?	element	
CoilBindingDetails ?	element	
CombBindingDetails ?	element	
RingBindingDetails ?	element	

8.63.1 Element: ChannelBindingDetails

Table 8-138: ChannelBindingDetails Element

Name	Data Type	Description
ClampD ?	double	The distance of the clamp that was “pressed away” (see Figure 8-46). Parameters used for channel binding. Note: @ClampD is for “ChannelBinding” only.
ClampSize ?	shape	The shape size of the clamp. The first number of the shape data type corresponds to the clamp width W (see Figure 8-46) which is determined by the final height of the block of Sheets to be bound. The second number corresponds to the length L (see Figure 8-46). The third corresponds to the spine length (not visible in Figure 8-46). The spine length is perpendicular on the paper plane. Note: @ClampSize is for “ChannelBinding” only.
Cover ?	boolean	If “true” the clamp is inside of a preassembled cover. Note: @Cover replaces 1.x @ClampSystem. Note: @Cover is for “ChannelBinding” only.

8.63.2 Element: CoilBindingDetails

Table 8-139: CoilBindingDetails Element

Name	Data Type	Description
Diameter ?	double	Specifies the diameter of the coil to be produced. The diameter is determined by the height of the block of Sheets to be bound.
Thickness ?	double	The thickness of the coil or comb. Note: @Thickness is for “CoilBinding”, “PlasticCombBinding” and “WireCombBinding” only.
Tucked ?	boolean	If “true”, the ends of the coils are “tucked in”. Note: @Tucked is for “CoilBinding” only.

8.63.3 Element: CombBindingDetails

Table 8-140: CombBindingDetails Element

Name	Data Type	Description
<i>Diameter?</i>	double	Specifies the diameter of the comb to be produced. The diameter is determined by the height of the block of Sheets to be bound
<i>Thickness?</i>	double	The thickness of the coil or comb. Note: @Thickness is for "CoilBinding", "PlasticCombBinding" and "WireCombBinding" only.
<i>WireCombShape?</i>	enumeration	The shape of the wire comb binding. Allowed values are: <i>Single</i> – Each “tooth” is made with one wire. <i>Twin</i> – The shape of each “tooth” is made with a double wire. Note: @WireCombShape replaces 1.x @Shape. Note: @WireCombShape is for "WireCombBinding" only.

8.63.4 Element: RingBindingDetails

Table 8-141: RingBindingDetails Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Diameter?</i>	double	Specifies the diameter of the rings, in points.
<i>RingMechanic?</i>	boolean	If "true", a hand lever is available for opening. Note: @RingMechanic is for "RingBinding" only.
<i>RingShape?</i>	NMTOKEN	<i>RingBinding</i> values: Values include: <i>Round</i> <i>Oval</i> <i>D-shape</i> <i>SlantD</i> Note: @RingShape is for "RingBinding" only.
<i>RivetsExposed?</i>	boolean	The following "RingBinding" choice describes mounting of ring mechanism in binder case. If "true", the heads of the rivets are visible on the exterior of the binder. If "false", the binder covering material covers the rivet heads. Note: @RivetsExposed is for "RingBinding" only.
<i>SpineWidth?</i>	double	The spine width is determined by the final height of the block of Sheets to be bound. Note: @SpineWidth is for "RingBinding" only.

Table 8-141: RingBindingDetails Element (Sheet 2 of 2)

Name	Data Type	Description
<i>ViewBinder</i> ?	NMTOKEN	<p>For "RingBinding" clear vinyl outer-wrap types on top of a colored base wrap:</p> <p>Values include:</p> <p><i>Embedded</i> – Printed material is embedded by sealing between the colored and clear vinyl layers during the binder manufacturing.</p> <p><i>Pocket</i> – Binder is designed so that printed material can be inserted between the color and clear vinyl layers after the binder is manufactured.</p> <p>Note: @<i>ViewBinder</i> is for "RingBinding" only.</p>

8.64 ManualLaborParams

This Resource describes the parameters to qualify generic manual work within graphic arts production. Additional Comment Elements will generally be needed to describe the work in human readable form.

Resource Properties

Resource referenced by: —

Input of Processes: **ManualLabor**

Output of Processes: —

Table 8-142: ManualLaborParams Resource

Name	Data Type	Description
<i>LaborType</i>	NMTOKEN	<p>Type of manual labor that is performed.</p> <p>Values include:</p> <p><i>CreateCoatingForm</i> – create a form to apply coatings during or after printing</p> <p><i>EditArt</i> – Unspecific Art editing (for work on specific files LayoutElementProduction is to be used)</p> <p><i>EditMarks</i> – Marks editing</p> <p><i>EditTraps</i> – Traps editing</p> <p><i>ManageJob</i> – General work on the Job.</p> <p><i>PhoneCallToCustomer</i> – Phone calls to ask/inform the Customer.</p> <p><i>SeparateBlanks</i> – Manual separation of blanks from a sheet after die cutting.</p>

8.65 Media

This Resource describes a physical element that represents a raw, unexposed printable surface such as Sheet, film or plate. @*Gloss*, @*MediaColorName* and @*Opacity* Attributes provide media characteristics pertinent to color management.

Properties

referenced by: **Color/**, **DieLayout**, **DigitalPrintingParams**, **EmbossingParams**/
Emboss, **ExposedMedia**, **FeedingParams/Feeder**,
FeedingParams/CollatingItem,
ImageSetterParams, **InterpretingParams**, **Layout**, **Media**/

Intent Pairing:	MediaLayers, RasterReadingParams, ShapeDef, StrippingParams, Tile
Input of Processes:	MediaIntent, HoleMakingIntent, ContentCheckIntent <i>Bending, BoxPacking, Bundling, CaseMaking, ConventionalPrinting, Cutting, DigitalPrinting, Embossing, Feeding, ImageSetting, Laminating, Varnishing, Wrapping</i>
Output of Processes:	<i>Cutting, Feeding</i>

Table 8-143: Media Resource (Sheet 1 of 9)

Name	Data Type	Description
<i>BackBrightness</i> ?	double	Equivalent to <i>@Brightness</i> (see below), but applied to the back surface of the Media. If not specified, the value of <i>@Brightness</i> applies to the Front and Back surfaces of the Media.
<i>BackCoatingDetail</i> ?	NMTOKEN	Identical to <i>@FrontCoatingDetail</i> (see below), but applied to the back surface of the media. Default value is from: <i>@FrontCoatingDetail</i> . Allowed values are from: <i>@FrontCoatingDetail</i> .
<i>BackCoatings</i> ?	enumeration	Identical to <i>FrontCoatings</i> (see below), but applied to the back surface of the media. Default value is from: <i>@FrontCoatings</i> . Allowed values are from: <i>@FrontCoatings</i> .
<i>BackGlossValue</i> ?	double	Gloss of the back surface of the media in gloss units as defined by [ISO8254-1:1999]. When not known, <i>@BackGlossValue</i> defaults to the value of <i>@FrontGlossValue</i> .
<i>BackGrade</i> ?	integer	Grade of the back side of the material. If not specified, defaults to the value of <i>@Grade</i> . See <i>@Grade</i> for details.
<i>BackISOPaperSubstrate</i> ?	integer	ISOPaperSubstate of the back side of the material. If not specified, defaults to the value of <i>@ISOPaperSubstrate</i> . See <i>@ISOPaperSubstrate</i> for details.
<i>Brightness</i> ?	double	Reflectance percentage of diffuse blue reflectance as defined by [ISO2470:1999]. The reflectance is reported per [ISO2470:1999] as the diffuse blue reflectance factor of the Media in percent to the nearest 0.5% reflectance factor. See also <i>@BackBrightness</i> .specified separately for the Front and Back surfaces by specifying both <i>@Brightness</i> and <i>@BackBrightness</i>
<i>CIELint</i> ?	double	Average CIE tint value. Average CIE tint is calculated according to equations given in [TAPPI T560].
<i>CIEWhiteness</i> ?	double	Average CIE whiteness value. Average CIE whiteness is calculated according to equations given in [TAPPI T560].
<i>CoreWeight</i> ?	double	Weight of the core of a Roll, in grams [g]

Table 8-143: Media Resource (Sheet 2 of 9)

Name	Data Type	Description
<i>Dimension</i> ?	XYPair	<p>The X and Y dimensions of the chosen medium, measured in points. <i>@Dimension</i> specifies the outer bounding box of the Media. The X, Y values of <i>@Dimension</i> establishes the user coordinate system into which content is mapped (i.e., the origin is in the lower left corner of the rectangle defined by "0 0 X Y"). In case of "<i>Roll</i>" media, the X coordinate specifies the reel width and the Y coordinate specifies the length of the Web in points. If a <i>@Dimension</i> coordinate is unknown, the value SHALL be "<i>0</i>". If not specified, the dimension is unknown. If either or both X or Y = "<i>0</i>" (i.e., unknown), the default orientation is assumed to be portrait (i.e., Y > X).</p> <p>Values include those from: Table G-1, "Media Sizes" on page 1193.</p>
<i>Flute</i> ?	NMTOKEN	<p>Single, capital letter that specifies the Flute type of corrugated media.</p> <p>Although the classification of flutes using a letter code "A", "B", etc., are used very frequently (e.g., in the specification of the order for a box), there seems to be no agreement on the exact numerical specification of those categories. Slightly varying numbers for flute size and frequency can be found between regions (European versus US) and between vendors.</p> <p>Values include:</p> <p>A B C</p>
<i>FluteDirection</i> ?	enumeration	<p>Direction of the fluting.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>LongEdge</i> – Along the longer axis as defined by <i>@Dimension</i>. <i>ShortEdge</i> – Along the shorter axis as defined by <i>@Dimension</i>. <i>XDirection</i> – Along the X-axis of the Media coordinate system <i>YDirection</i> – Along the Y-axis of the Media coordinate system
<i>FrontCoatingDetail</i> ?	NMTOKEN	<p>Describes (beyond <i>@FrontCoatings</i>) the coating to the front surface of the media and possibly the technology used to apply the coating.</p> <p>Values include:</p> <p><i>Cast</i></p>

Table 8-143: Media Resource (Sheet 3 of 9)

Name	Data Type	Description
<i>FrontCoatings</i> ?	enumeration	<p>What preprocess coating has been applied to the front surface of the media.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>None</i> – No coating. <i>Coated</i> – A coating of a system-specified type. <i>Glossy</i> <i>HighGloss</i> <i>Matte</i> <i>Polymer</i> – Coating for a photo polymer process <i>Silver</i> – Coating for a silver halide process <i>Satin</i> <i>Semigloss</i>
<i>FrontGlossValue</i> ?	double	Gloss of the front side of the media in gloss units as defined by [ISO8254-1:1999]. Refer also to [TAPPI T480] for examples of gloss calculation.
<i>Grade</i> ?	integer	<p>The <i>@Grade</i> of the media on a scale of 1 through 5. The <i>@Grade</i> is ignored if <i>@MediaType</i> is not "Paper". <i>@Grade</i> of paper material is defined in accordance with the paper "types" set forth in [ISO12647-2:2004]. If a workflow supports <i>@ISOPaperSubstrate</i>, and both <i>@Grade</i> and <i>@ISOPaperSubstrate</i> are present, it SHALL use <i>@ISOPaperSubstrate</i>.</p> <p>Note: [ISO12647-2:2004] paper type Attribute Values do NOT align with U.S. GRACOL paper grade Attribute Values (e.g., [ISO12647-2:2004] type 1 does not equal U.S. GRACOL grade 1).</p> <p>The values define offset printing paper types.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> 1 – Gloss-coated paper. 2 – Matt-coated paper. 3 – Gloss-coated, Web paper. 4 – Uncoated, white paper. 5 – Uncoated, yellowish paper.
<i>GrainDirection</i> ?	enumeration	<p>Direction of the grain in the coordinate system defined by <i>@Dimension</i>.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>XDirection</i> – Along the X-axis of the Media coordinate system. <i>YDirection</i> – Along the Y-axis of the Media coordinate system.

Table 8-143: Media Resource (Sheet 4 of 9)

Name	Data Type	Description
<i>HoleType</i> ?	enumerations	<p>Predefined hole pattern. Multiple hole patterns are allowed (e.g., 3-hole ring binding and 4-hole ring binding holes on one piece of media). For details of the hole types, refer to Appendix M, “JDF XJDF/CIP4 Hole Pattern Catalog” on page 1271.</p> <p>Allowed values are:</p> <p><i>None</i> – No holes.</p> <p><i>Explicit</i> – Holes are defined in a <i>HolePattern</i>.</p> <p>Allowed values are from: Appendix M, “JDF XJDF/CIP4 Hole Pattern Catalog” on page 1271.</p>
<i>ImagableSide</i> ?	enumeration	<p>Side of the chosen medium that are to be marked.</p> <p>Allowed values are:</p> <p><i>Front</i></p> <p><i>Back</i></p> <p><i>Both</i></p> <p><i>Neither</i></p>
<i>InnerCoreDiameter</i> ?	double	Specifies the inner diameter of the core of a Roll, in points. See also <i>@OuterCoreDiameter</i> and <i>@RollDiameter</i> .
<i>InsideLoss</i> ?	double	The inside loss of corrugated board material in microns [μm]. Note: <i>@InsideLoss</i> + <i>@OutsideGain</i> need not be exactly equal to thickness.
<i>ISOPaperSubstrate</i> ?	enumeration	<p>The Paper Substrate Type of the Medium from "PS1" through "PS8".</p> <p><i>@ISOPaperSubstrate</i> supersedes <i>@Grade</i> and adds new values to allow for improved papers.</p> <p>If a workflow supports <i>@ISOPaperSubstrate</i>, and both <i>@Grade</i> and <i>@ISOPaperSubstrate</i> are present, it SHALL use <i>@ISOPaperSubstrate</i>.</p> <p><i>@ISOPaperSubstrate</i> type of paper material is defined in accordance with the Print Substrate set forth in [ISO12647-2:2013].</p> <p>Allowed values are:</p> <p><i>PS1</i> – Premium Coated</p> <p><i>PS2</i> – Improved Coated</p> <p><i>PS3</i> – Standard Coated Glossy</p> <p><i>PS4</i> – Standard Coated Matte</p> <p><i>PS5</i> – Wood-free Uncoated</p> <p><i>PS6</i> – Super Calendered</p> <p><i>PS7</i> – Improved Uncoated</p> <p><i>PS8</i> – Standard Uncoated</p>
<i>LabColorValue</i> ?	LabColor	<i>@LabColorValue</i> is the CIELAB color value of the media, computed as specified in [TAPPI T527].

Table 8-143: Media Resource (Sheet 5 of 9)

Name	Data Type	Description
<i>MediaColorName</i> ?	NamedColor	A name for the color. If more specific, specialized or site-defined media color names are needed, use <i>@MediaColorNameDetails</i> .
<i>MediaColorNameDetails</i> ?	string	A more specific, specialized or site-defined name for the media color. If <i>@MediaColorNameDetails</i> is supplied, <i>@MediaColorName</i> SHOULD also be supplied. <i>@MediaColorNameDetails</i> SHOULD be used to specify specialized media properties such as holographic or transparent foils.
<i>MediaQuality</i> ?	string	Named quality description of the media. For folding carton quality, multiple named quality description systems are in use (e.g., GC1, SBB, etc.). For an overview, see http://www.procarton.com/files/fact_file_6.pdf
<i>MediaSetCount</i> ?	integer	When the input media is grouped in sets, identifies the number of pieces of media in each set. For example, if the <i>@MediaTypeDetails</i> is "PreCutTabs", a <i>@MediaSetCount</i> of "5" would indicate that each set includes five tab Sheets.
<i>MediaType</i> ?	NMTOKEN	<p>Describes the medium being employed.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>CorrugatedBoard</i> <i>Disc</i> – CD or DVD disc to be printed on. <i>EndBoard</i> – end board used in the Bundling Process. <i>EmbossingFoil</i> <i>Film</i> <i>Foil</i> <i>GravureCylinder</i> – gravure cylinder. <i>ImagingCylinder</i> – reusable direct imaging cylinder in a press. <i>LaminatingFoil</i> <i>MountingTape</i> – for flexo plate mounting tape. <i>Other</i> – not one of the defined values. <i>Paper</i> <i>Plate</i> <i>Screen</i> – used for Screen Printing. <i>SelfAdhesive</i> <i>Sleeve</i> – for flexo sleeves <i>ShrinkFoil</i> <i>Textile</i> <i>Transparency</i> <i>Vinyl</i>

Table 8-143: Media Resource (Sheet 6 of 9)

Name	Data Type	Description
<i>MediaTypeDetails</i> ?	NMTOKEN	<p>Additional details of the chosen medium.</p> <p>Constraint: If <i>@MediaTypeDetails</i> is specified, <i>@MediaType</i> SHALL be specified.</p> <p>Values include those from: Table 8-195, “<i>MediaTypeDetails</i> Attribute Values” on page 812.</p>
<i>MediaUnit</i> ?	enumeration	<p>Describes the format of the media as it is delivered to the Device.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Continuous</i> – Continuously connected Sheets which can be fan folded. <i>Roll</i> <i>Sheet</i> – Individual cut Sheets.
<i>Opacity</i> ?	enumeration	<p>The opacity of the media. See <i>@OpacityLevel</i> to specify the degree of opacity for any of these values.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Opaque</i> – The media is opaque. With two-sided printing the printing on the other side does not show through under normal incident light. <i>Translucent</i> – The media is translucent to a system specified amount. For example, translucent media can be used for back lit viewing. <i>Transparent</i> – The media is transparent.
<i>OpacityLevel</i> ?	double	Normalized TAPPI opacity (Cn), as defined and computed in [ISO2471:1998]. Refer also to [TAPPI T519] for calculation examples.
<i>OuterCoreDiameter</i> ?	double	Specifies the outer diameter of the core of a Roll, in points. See also <i>@InnerCoreDiameter</i> and <i>@RollDiameter</i> .
<i>OutsideGain</i> ?	double	The outside gain of corrugated board material in microns [μm].
<i>PlateTechnology</i> ?	enumeration	<p>Exposure technology of the plates.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>FlexoAnalogSolvent</i> ⁴ <i>FlexoAnalogThermal</i> <i>FlexoDigitalSolvent</i> <i>FlexoDigitalThermal</i> <i>FlexoDirectEngraving</i> <i>InkJet</i> – Exposure with inkjet technology. Note that <i>@FrontCoatings = "Inkjet"</i> specifies inkjet specific coating of paper or transparency Media, not of plates. <i>Thermal</i> – Thermal exposure <i>UV</i> – Ultraviolet exposure <i>Visible</i> – Visible light exposure

Table 8-143: Media Resource (Sheet 7 of 9)

Name	Data Type	Description
<i>Polarity</i> ?	enumeration	<p>Polarity of the chosen medium.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Positive</i> <i>Negative</i>
<i>PrintingTechnology</i> ?	NMTOKEN	<p>Describes the printing technology that the media or coatings on the media are intended for or optimized for.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>DyeSublimation</i> <i>Electrostatic</i> <i>InkJet</i> <i>Laser</i> <i>Latex</i> <i>Offset</i> <i>Thermal</i> <i>UV</i>
<i>RecycledPercentage</i> ?	double	The percentage, between 0 and 100, of recycled material that the media is to contain.
<i>ReliefThickness</i> ?	double	The thickness of the relief, measured in microns [μm]. The floor thickness can be calculated as (@Thickness - @ReliefThickness). See Figure 8-48.
<i>RollDiameter</i> ?	double	Specifies diameter of a Roll, in points. See also @InnerCoreDiameter and @OuterCoreDiameter.
<i>ShrinkIndex</i> ?	XYPair	Specifies the ratio of the media linear dimension after shrinking to prior shrinking. The X Value specifies index in the major shrink axis, whereas the Y Value specifies the index in the minor shrink axis. Used to describe shrink wrap media.
<i>SleeveInterlock</i> ?	NMTOKEN	<p>The type of interlock (or notch) to use for a flexo sleeve.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Type01</i> – see Figure 8-49. ... <i>Type20</i> – see Figure 8-49.

Table 8-143: Media Resource (Sheet 8 of 9)

Name	Data Type	Description
<i>StockType</i> ?	NMTOKEN	<p>Strings describing the available stock.</p> <p>@<i>StockType</i> defines the base size when calculating North American or Japanese paper weights. See Appendix F, “North American and Japanese Media Weight Explained” on page 1189.</p> <p>Values with Kanji names support Japanese media.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Bible</i> – equivalent to “<i>Book</i>”. <i>Book</i> – Base size: 25" x 38" <i>Bond</i> – Base size: 17" x 22" <i>Bristol</i> – Base size: 22½" x 28½" <i>Coated</i> – equivalent to “<i>Book</i>”. <i>Cover</i> – Base size: 20" x 26" <i>Index</i> – Base size: 25½" x 30½" <i>Ledger</i> – equivalent to “<i>Bond</i>”. <i>Manifold</i> – equivalent to “<i>Bond</i>”. <i>Newsprint</i> – Base size: 24" x 36" <i>Offset</i> – This includes book stock, equivalent to “<i>Book</i>”. <i>Tag</i> – equivalent to “<i>Newsprint</i>”. <i>Text</i> – equivalent to “<i>Book</i>”. <i>Aatoposutoshi</i> – アートポスト紙 (“art-post paper”) is cover stock coated on one side. <i>Aatoshi</i> – アート紙 (“art paper”) is machine coated paper, available in top quality and medium quality (Joushitsu and Chuushitsu). <i>Chuushitsushi</i> – 中質紙 (“medium-quality paper”) contains a minimum of 70% chemical pulp. <i>Joushitsushi</i> – 上質紙 (“top-quality paper”) contains 100% chemical pulp. <i>Mashinkootoshi</i> – マシンコート紙 (“machine coated paper”), also called Kootoshi (コート紙), is machine coated paper given only a thin coat of clay.

Table 8-143: Media Resource (Sheet 9 of 9)

Name	Data Type	Description
<i>Texture</i> ?	NMTOKEN	<p>The intended texture of the media.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Antique</i> – Rougher than vellum surface. <i>Calendared</i> – Extra smooth or polished, uncoated paper. <i>Glossy</i> – Glossy Media, e.g. laminating foil <i>Linen</i> – Texture of coarse woven cloth. <i>Matte</i> – Matte Media, e.g. laminating foil <i>Smooth</i> <i>Stipple</i> – Fine pebble finish. <i>Uncalendared</i> – Rough, unpolished and uncoated papers. <i>Vellum</i> – Slightly rough surface.
<i>Thickness</i> ?	double	<p>The thickness of the chosen medium, measured in microns [μm].</p> <p>Note: Thickness is often referred to as caliper.</p>
<i>Weight</i> ?	double	Weight of the chosen medium, measured in grams per square meter [g/m^2]. See Appendix F, “North American and Japanese Media Weight Explained” on page 1189 for details on converting North American paper weights to g/m^2 .
<i>WrapperWeight</i> ?	double	Weight of the wrapper of a Roll, in grams [g]
<i>HolePattern</i> *	element	Explicit list of holes. <i>HolePattern</i> SHALL be specified if <i>@HoleType</i> = "Explicit".
<i>MediaLayers</i> ?	element	Subelement describing the layer structure of media such as corrugated or self adhesive materials.
<i>TabDimensions</i> ?	element	<p>Specifies the dimensions of the tabs when <i>@MediaTypeDetails</i> = "TabStock", "PreCutTabs" or "FullCutTabs".</p> <p>Note: see BindingIntent/Tabs (Table 7-42, “Tabs Element” on page 472) (rather than MediaIntent) for how tabbed media is specified in Product Intent.</p>

— Attribute: **MediaTypeDetails**

Table 8-144: MediaTypeDetails Attribute Values (Sheet 1 of 2)

Value	Description
<i>Aluminum</i>	Conventional or CtP press plate.
<i>Backlit</i>	Any Media that is designed to be illuminated from the back side.
<i>Cardboard</i>	
<i>CD</i>	CD disc to be printed on.

Table 8-144: MediaTypeDetails Attribute Values (Sheet 2 of 2)

Value	Description
<i>ContinuousLong</i>	Continuously connected Sheets of an opaque material connected along the long edge.
<i>ContinuousShort</i>	Continuously connected Sheets of an opaque material connected along the short edge.
<i>DoubleWall</i>	Double wall corrugated board
<i>DVD</i>	DVD disc to be printed on.
<i>DryFilm</i>	
<i>Envelope</i>	Envelopes that can be used for conventional mailing purposes.
<i>EnvelopePlain</i>	Envelopes that are not preprinted and have no windows.
<i>EnvelopeWindow</i>	Envelopes that have windows for addressing purposes.
<i>FlexoBase</i>	For the base layer of flexo plates.
<i>FlexoPhotoPolymer</i>	For the photopolymer layer of flexo plates.
<i>Flute</i>	Flute layer of a corrugated board
<i>FullCutTabs</i>	Media with a tab that runs the full length of the medium so that only one tab is visible extending out beyond the edge of non-tabbed media.
<i>ImageSetterPaper</i>	Contact paper as replacement for film.
<i>Labels</i>	Label stock (e.g., a Sheet of peel-off labels).
<i>Letterhead</i>	Separately cut Sheets of an opaque material including a letterhead.
<i>MultiLayer</i>	Form medium composed of multiple layers which are preattached to one another (e.g., for use with impact printers).
<i>MultiPartForm</i>	Form medium composed of multiple layers not preattached to one another; each Sheet might be drawn separately from an input source.
<i>Photographic</i>	Separately cut Sheets of an opaque material to produce photographic quality images.
<i>Polyester</i>	Conventional or CtP press plate.
<i>PreCutTabs</i>	Media with tabs that are cut so that more than one tab is visible extending out beyond the edge of non-tabbed media.
<i>ScrimBanner</i>	Specific type of vinyl.
<i>SingleFace</i>	Single face corrugated board.
<i>SingleWall</i>	Single wall corrugated board.
<i>Stationery</i>	Separately cut Sheets of an opaque material, includes generic paper.
<i>TabStock</i>	Media with tabs, either precut or full-cut.–
<i>Tractor</i>	Tractor feed with holes.
<i>TripleWall</i>	Triple wall corrugated board
<i>WallPaper</i>	Details of Paper.
<i>WetFilm</i>	Conventional photographic film.

Figure 8-38: Paper Roll with some Roll-specific Information

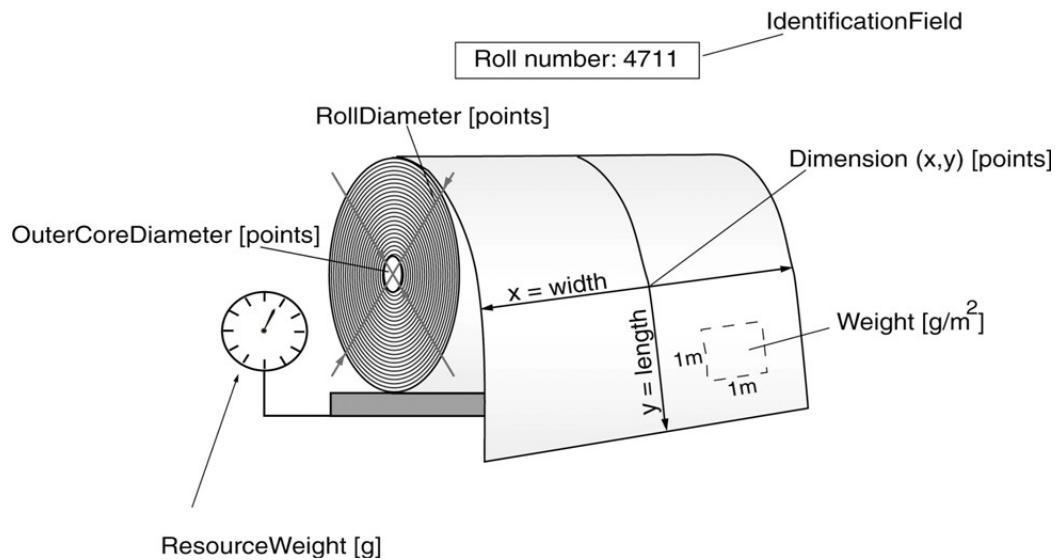


Figure 8-39: Relief and Floor Thickness for a Flexo Plate or Flexo Sleeve

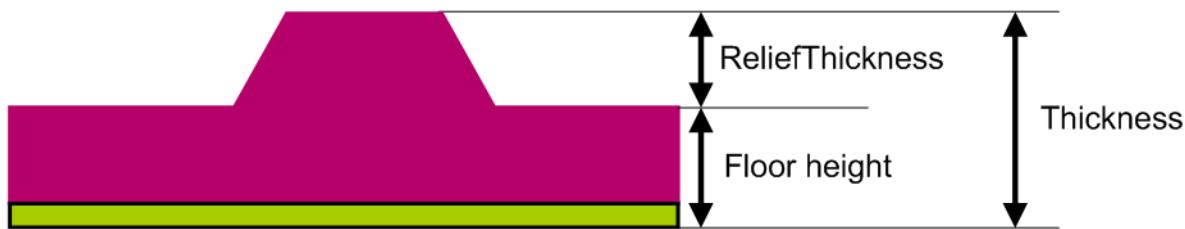
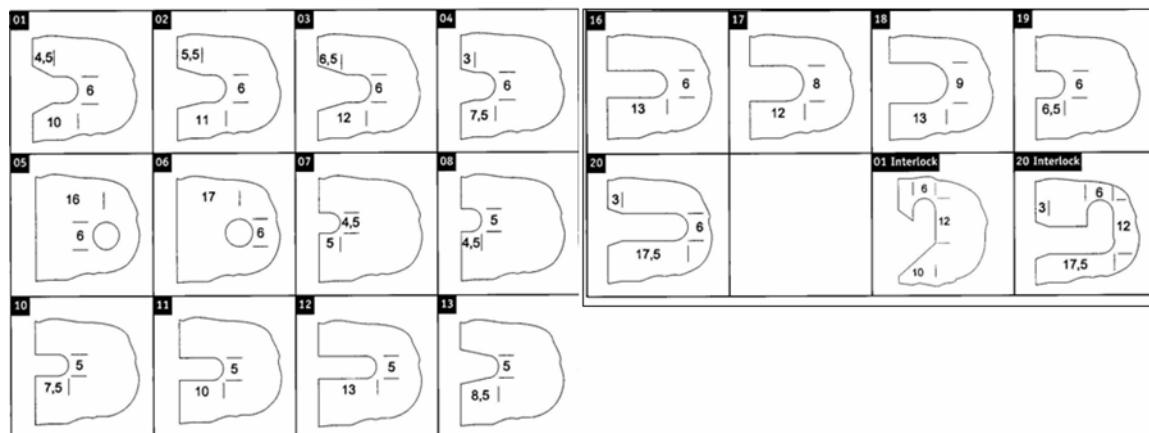


Figure 8-40: Types of Interlocks for Flexo Sleeve



8.65.1 Element: TabDimensions

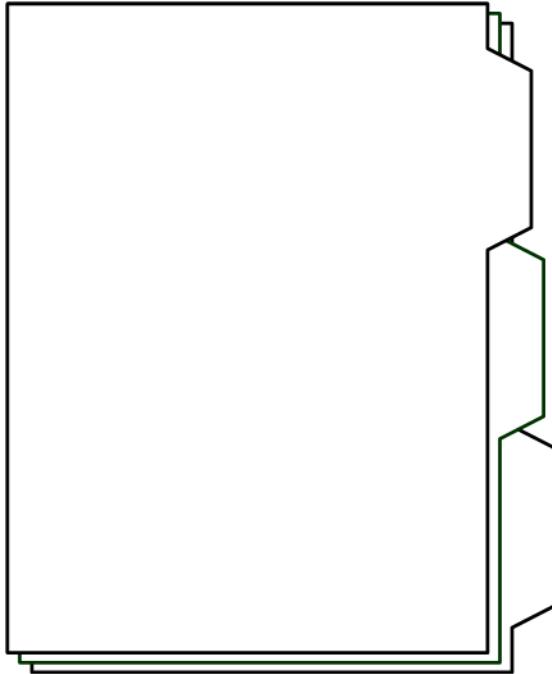
Specifies the size and placement of tabs in a bank and in a set of tab stock.

Table 8-145: TabDimensions Element

Name	Data Type	Description
<i>TabEdge</i> ?	enumeration	<p>Indicates which edge of the media has tabs. Sets the coordinate system for <i>@TabOffset</i>, <i>@TabExtensionDistance</i>, and <i>@TabWidth</i>.</p> <p>Allowed values are:</p> <p><i>Left</i> <i>Top</i> <i>Right</i> <i>Bottom</i></p>
<i>TabExtensionDistance</i> ?	double	<p>The positive distance in points that the tab extends beyond the body of the other media.</p> <p>Note: same as BindingIntent/Tabs/@TabExtensionDistance.</p> <p>Note: This value is always included in the value of the overall extent of the Media defined by Media/@Dimension. See Figure 8-51, “Diagram of a Single Bank of Tabs,” on page 817.</p>
<i>TabOffset</i> ?	double	<p>Specifies the magnitude of the distance in points from the two corners to the edge of the first “tab pitch” point of the first tab in the bank along the <i>@TabEdge</i>. This distance is the same on both ends of the bank of tabs. See Figure 8-51, “Diagram of a Single Bank of Tabs,” on page 817.</p>
<i>TabSetCollationOrder</i> ?	NMTOKEN	<p>Collation order of media provided in sets. Applicable to sets of pre-cut tabs. See Figure 8-50, “TabSetCollationOrder Attribute Values,” on page 816.</p> <p>Values include:</p> <p><i>Forward</i> – first tab is towards top of stack <i>Reverse</i> – first tab is toward bottom of stack.</p>
<i>TabsPerBank</i> ?	integer	<p>Specifies the number of equal-sized tabs in a single bank if all positions were filled.</p> <p>Note: banks can have tabs only in some of the possible positions.</p> <p>Note: same as BindingIntent/Tabs/@TabsPerBank.</p> <p>Media/@MediaSetCount specifies the number of tabs per set. A set can consist of one or more banks. If Media/@MediaSetCount is not an even multiple of <i>@TabsPerBank</i>, the last bank in each set is partially filled.</p>
<i>TabWidth</i> ?	double	<p>The width along the <i>@TabEdge</i> of each tab as measured along the mid-line of the tab. Each tab is centered within a space called the “tab pitch”. See Figure 8-51, “Diagram of a Single Bank of Tabs,” on page 817.</p>

Figure 8-41: TabSetCollationOrder Attribute Values

TabSetCollationOrder = "Forward"



TabSetCollationOrder = "Reverse"

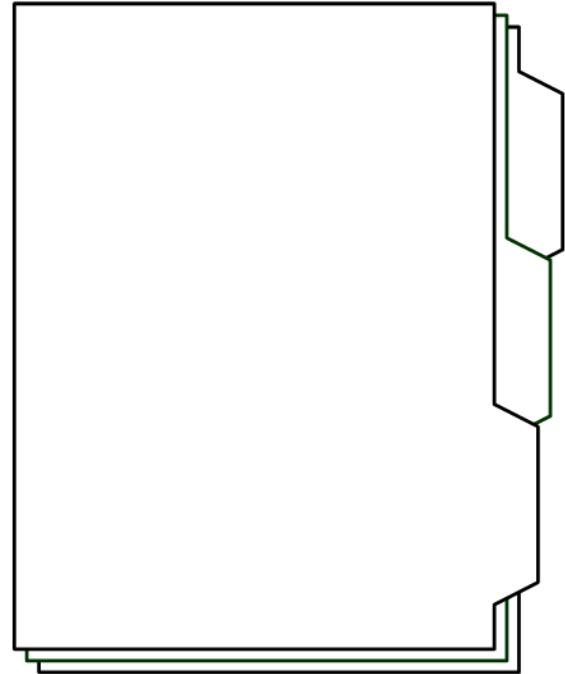
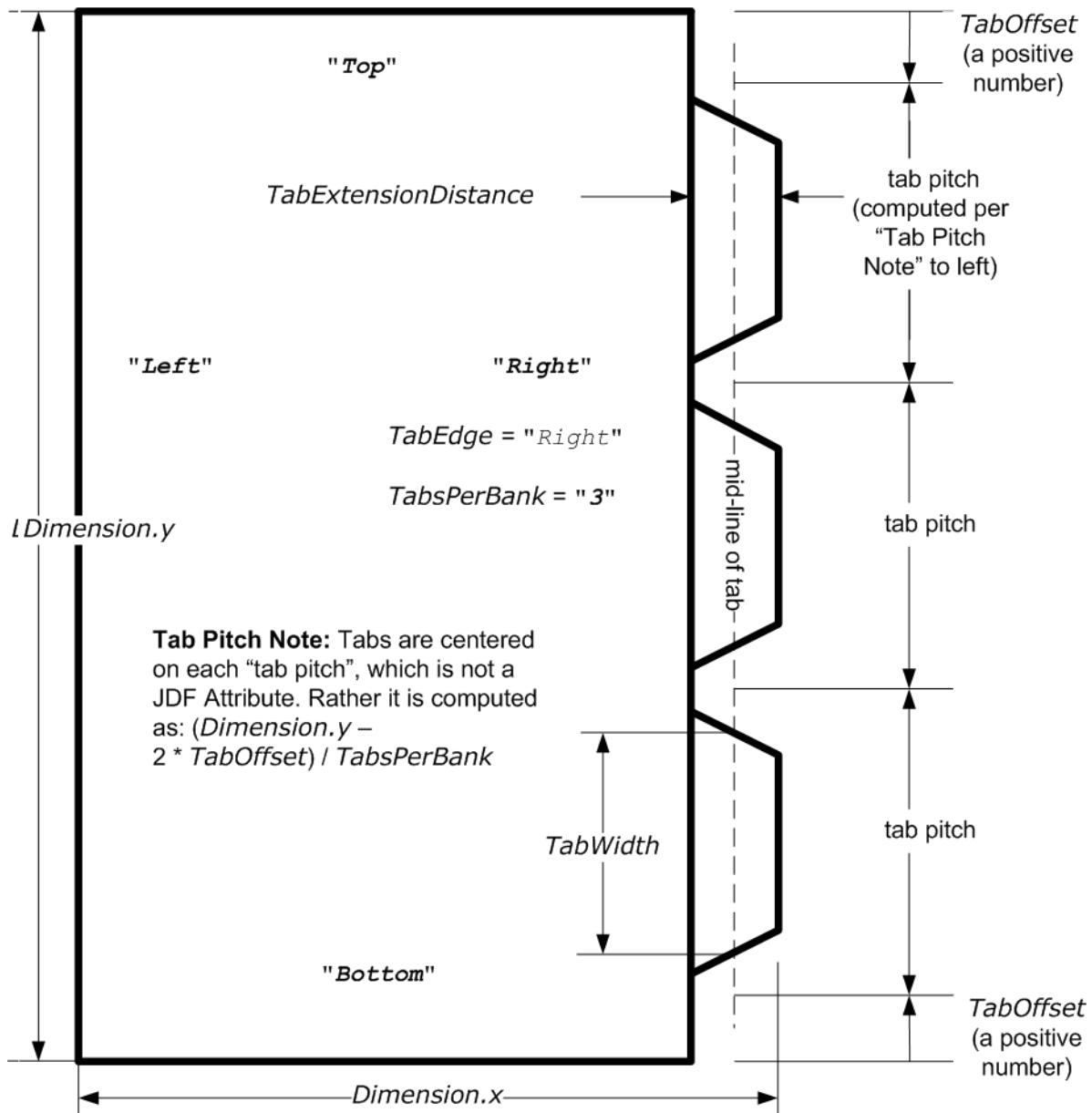


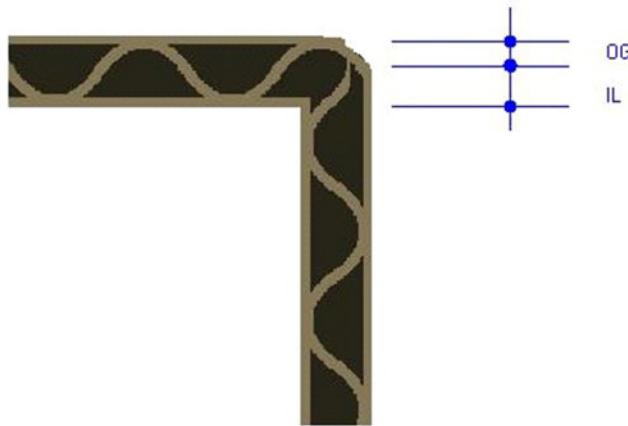
Figure 8-42: Diagram of a Single Bank of Tabs



8.65.2 More about Media

8.65.2.1 Inside Loss and Outside Gain

Inside loss and outside gain: dimensional values used in the mechanical design phase of a box. Note: IL + OG is not exactly equal to thickness. Thickness is most often referred to as caliper.

Figure 8-43: Inside Loss, Outside Gain

8.65.2.2 Corrugated Media:

Corrugated material consists of multiple Sheets of paper (called liners) with fluted material in between. For background information on Corrugated Media, see <http://cpc.corrugated.org/Basics>. Corrugated media comes in different variants.

- Number of layers:
 - single face (1 liner, 1 flute),
 - single wall (2 liners, 1 flute),
 - double wall (3 liners, 2 flutes),
 - triple wall (4 liners, 3 flutes)
- Flute size and frequency: A, B, C, E, F flute. See <http://cpc.corrugated.org/Basics/BasicAllAbout.aspx>

Example 8-23: Media: Corrugated

TBD 2.x Example.

```
<Media Class="Consumable" ID="M123456" ProductID="B190Y180D1050x120"
      Status="Available" DescriptiveName="B Flute 190Y 180D 1050x1210"
      Dimension="1050.0 120.0" MediaType="CorrugatedBoard"
      MediaTypeDetails="SingleWall" MediaUnit="Sheet" Thickness="2382.0"
      InsideLoss="1000.0" OutsideGain="1380.0" Weight="600">
<MediaLayers>
  <!-- FrontLiner -->
  <Media DescriptiveName="190gsm clay coated" MediaType="Paper"
        Weight="190" FrontCoatings="Coated"/>
  <!-- Flute -->
  <Media DescriptiveName="Flute" MediaType="Paper" Weight="180"
        FluteDirection="ShortEdge" Flute="B" MediaTypeDetails="Flute"/>
  <!-- BackLiner -->
  <Media DescriptiveName="180gsm white top" MediaType="Paper"
        Weight="180"/>
</MediaLayers>
</Media>
```

8.65.2.3 Self adhesive Media

Self adhesive media is described as **MediaLayers** Elements with nested **Media** and **GlueLine** Elements.

Example 8-24: Media: Self Adhesive

TBD 2.x Example.

```
<Media Class="Consumable" ID="M123456" ProductID="7890123" Status="Available"
      DescriptiveName="40# Fasson coated label stock" Dimension="1134.0 0"
      MediaType="SelfAdhesive" MediaUnit="Roll" Thickness="1000.0"
      Weight="150">
  <MediaLayers>
    <!-- Front -->
    <Media DescriptiveName="Antique Cream Smooth WS IL" MediaType="Paper"
           Weight="90"/>
    <!-- Glue -->
    <GlueLine DescriptiveName="Permanent 91A" AreaGlue="true"
               GlueType="Hotmelt" GlueBrand="Uhu"/>
    <!-- Back -->
    <Media DescriptiveName="Blue Glassine 50" MediaType="Paper" Weight="50"/>
  </MediaLayers>
</Media>
```

8.65.2.4 Flexo Plate Media

A sample of a flexo plate with dimensions of 900 mm x 1200 mm, a base of 177 microns and a total thickness of 1143 microns.

A raw plate can contain several separations from multiple jobs. The real printing dimensions can only be determined when all elements of the mounting process are known: circumference of the sleeve on which the flat plate will be mounted, thickness of the mounting tape, thickness of base and thickness of the photopolymer.

Example 8-25: Media: Flat Plate

TBD 2.x Example.

```
<Media Class="Consumable" ID="M123456" ProductID="FlexoPlate"
      Status="Available" DescriptiveName="" Dimension="2551.181 3401.574"
      MediaType="Plate" PlateTechnology="FlexoDigitalThermal"
      Manufacturer="PlateManufacturerA" Brand="BrandB" BatchID="Batch 12345"
      Thickness="1143" ReliefThickness="500">
  <!--MediaLayers contains 2 items: the base layer and the
      photopolymer layer of the flexo plate -->
  <MediaLayers>
    <Media DescriptiveName="Base" MediaType="Plate"
           MediaTypeDetails="FlexoPlateBase" Thickness="177"/>
    <Media DescriptiveName="Photopolymer Layer" MediaType="Plate"
           MediaTypeDetails="FlexoPlatePhotopolymer" Thickness="966"/>
  </MediaLayers>
</Media>
```

8.65.2.5 Flexo Sleeve Media

The flexo sleeve has dimensions of 500 x 250 mm, a base of 1249 microns and a total thickness of 2810 microns. The sleeve dimensions are identical to printing dimensions (no distortion).

Example 8-26: Media: Flexo Sleeve

TBD 2.x Example.

```
<Media Class="Consumable" ID="M123456" ProductID="FlexoSleeve"
      Status="Available"
```

```

DescriptiveName="Sleeve" Dimension"1417.32 750.0" Media Type="Sleeve"
PlateTechnology="FlexoDigitalSolvent" Manufacturer="PlateManufacturerB"
Brand="BrandB" BatchID="Batch 6789" Thickness="2810"
ReliefThickness="500">
<!--MediaLayers contains 2 items: the base layer and the
photopolymer layer of the flexo plate -->
<MediaLayersMedia DescriptiveName="Base" Media Type="Plate"
    MediaTypeDetails="FlexoPlateBase" Thickness="1249"/>
  <Media DescriptiveName="Photopolymer Layer" Media Type="Plate"
    MediaTypeDetails="FlexoPhotopolymer" Thickness="1570"/>
</MediaLayersMedia>

```

8.66 MiscConsumable

The **MiscConsumable** Resource is intended for cost accounting, inventory control and availability scheduling of supplies used in the production workflow where a more detailed parameterization of the Resource is not necessary. **MiscConsumable** is limited to modeling consumables not already more specifically defined in JDF (**Ink**, **Media**, **Pallet**, **RegisterRibbon**, **Strap** or **UsageCounter**).

MiscConsumable Resources MAY appear as inputs to any **XJDF** Process. The default Unit for Amounts of **MiscConsumable** is Countable Objects.

Certain types of **MiscConsumable** Elements such as **MiscConsumable[@ConsumableType = "WasteContainer"]** are typically “consumed” by being filled. The sense of the **@Amount** Attribute for such Resources shall be the quantity of unused or empty waste containers that are available. If **@Unit** is a volume, distance or weight instead of Countable Objects, such **@Amount** will still represent the remaining unused capacity of the waste container.

Resource Properties

Resource referenced by: —

Input of Processes: **Any Process**

Output of Processes: —

Table 8-146: MiscConsumable Resource

Name	Data Type	Description
Type ?	NMTOKEN	<p>Identifies the type of MiscConsumable (machine-readable). A human-readable (possibly localized) description of the consumable SHOULD also be supplied in <i>@DescriptiveName</i>.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Developer</i> – Chemicals used in filmsetters and platesetters. <i>Electricity</i> – Electrical energy. Typically monitored for CO₂ tracking. Measured in kWh. <i>FuserOil</i> – Silicon Oil <i>Gas</i> – Natural Gas. Typically monitored for CO₂ tracking. Measured in m³. <i>Glue</i> <i>Headband</i> <i>RegistrationRibbon</i> <i>Paperclips</i> <i>Staples</i> <i>WasteContainer</i> – Waste Toner Bottle. <i>Wire</i> – bulk wire used for forming staples or other binding.
TypeDetails ?	string	Additional site specific details of the consumable.

8.67 NodeInfo

The **NodeInfo** Resource contains information about planned scheduling. It allows MIS to plan, schedule and invoice Jobs or Job Parts.

Resource Properties

Resource referenced by: —

Input of Processes: *Any Process*

Output of Processes: —

Table 8-147: NodeInfo Resource (Sheet 1 of 2)

Name	Data Type	Description
CleanupDuration ?	duration	Estimated duration of the clean-up phase of the Process.
DueLevel ?	enumeration	<p>Description of the severity of a missed deadline.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Trivial</i> – Missing the deadline has minor or no consequences. <i>Penalty</i> – Missing the deadline incurs a penalty. <i>JobCancelled</i> – The Job is cancelled if the deadline is missed.
End ?	dateTime	Date and time at which the Process is scheduled to end.
FirstEnd ?	dateTime	Earliest date and time at which the Process is to end.
FirstStart ?	dateTime	Earliest date and time at which the Process is to begin.

Table 8-147: NodeInfo Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>JobPriority</i> ?	integer	The scheduling priority for the Node where 100 is the highest and 0 is the lowest. Amongst the Nodes that can be processed in the XJDF Instance, all higher priority Nodes are to be processed before any lower priority ones. If one or more of the deadline oriented Attributes (e.g., <i>@FirstStart</i> or <i>@LastEnd</i>) is specified, such attribute(s) SHALL be honored before considering <i>@JobPriority</i> . The priority from JMF (QueueSubmissionParams/ <i>@Priority</i> or ModifyQueueEntryParams/ <i>@Priority</i>) takes precedence over NodeInfo / <i>@JobPriority</i> .
<i>LastEnd</i> ?	dateTime	Latest date and time at which the Process is to end. This is the deadline to which <i>@DueLevel</i> refers.
<i>LastStart</i> ?	dateTime	Latest date and time at which the Process is to begin.
<i>NaturalLang</i> ?	language	Language selected for communicating Attributes. If not specified, the operating system language is assumed.
<i>NodeStatus</i> ?	enumeration	Identifies the status of an individual part of the Node. Allowed values are from: Table 8-199, “NodeStatus Attribute Values” on page 824.
<i>NodeStatusDetails</i> ?	string	Machine readable description of the status that provides details beyond the enumerative values given by <i>@NodeStatus</i> . Values include those from: Section C.1, “StatusDetails Supported Strings” on page 1165.
<i>SetupDuration</i> ?	duration	Estimated duration of the setup phase of the Process.
<i>Start</i> ?	dateTime	Date and time of the planned Process start.
<i>TotalDuration</i> ?	duration	Estimated total duration of the Process, including setup and cleanup.
<i>WorkStepID</i> ?	NMTOKEN	ID of an individual work step (e.g., a Press Run). If NodeInfo is not Partitioned, or all Partitions are executed simultaneously, <i>@WorkStepID</i> corresponds to <i>@JobPartID</i> .
<i>Contact</i> ?	element	The internal administrator or supervisor that is responsible for the product or Process defined in this Node.
<i>MISDetails</i> ?	element	Definition how the costs for the execution of this Node are to be charged.
<i>NotificationFilter</i> *	element	Defines the set of NotificationFilter Elements that are to be logged in the AuditPool . This provides a logging method for Devices that do not support XJMF messaging. For details of the NotificationFilter Element, see Section 5.8.7.1.1, “NotificationFilter”.

— Attribute: NodeStatus**Table 8-148: NodeStatus Attribute Values (Sheet 1 of 2)**

Value	Description
<i>Aborted</i>	Indicates that the Process executing the Node has been aborted, which means that execution will not be resumed again.
<i>Cleanup</i>	The Process represented by this Node is currently being cleaned up.
<i>Completed</i>	Indicates that the Node or queue entry has been executed correctly, and is finished.
<i>InProgress</i>	The Node is currently executing.

Table 8-148: NodeStatus Attribute Values (Sheet 2 of 2)

Value	Description
<i>Setup</i>	The Process represented by this Node is currently being set up.
<i>Stopped</i>	Execution has been stopped. If a Job is " <i>Stopped</i> ", running can be resumed later. This status can indicate a break, a pause, maintenance or a breakdown — in short, any pause that does not lead the Job to be aborted.
<i>Suspended</i>	Execution has been stopped. If a Job is " <i>Suspended</i> ", running will be resumed later. Unlike " <i>Stopped</i> " this status indicates that the Job has been taken off the Device to execute another Job or perform some other action that is not related to this Job. When resumed, the Job MAY go into <i>@NodeStatus</i> = " <i>Setup</i> " before changing to " <i>InProgress</i> " again. The value " <i>Suspended</i> " is also used to describe iterations. In an iterative environment, " <i>Suspended</i> " specifies that at least one iteration cycle has completed but additional iteration cycles MAY still occur. In this use case, <i>@NodeStatusDetails</i> SHOULD be set to " <i>IterationPaused</i> "
<i>Waiting</i>	The Node can be executed.

ExposedMedia

8.68 Pallet

A **Pallet** represents the pallet used in packing goods.

Resource Properties

Resource referenced by: —

Input of Processes: **Palletizing**

Output of Processes: —

Table 8-149: Pallet Resource

Name	Data Type	Description
<i>PalletType</i>	NMTOKEN	<p>Type of pallet used.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>2Way</i> – Two-way entry. <i>4Way</i> – Four-way entry. <i>Euro800x600</i> – 800x600mm according to 15146-4 (equals half Euro pallet). <i>Euro800x1200</i> – 800x1200mm according to DIN EN 13698-1 (equals Euro pallet). <i>Euro1000x1200</i> – 1000x1200mm according to DIN EN 13698-2 (flat pallet). <i>Euro1200x1200</i> – 1200x1200mm no norm, but in use in the field.
<i>Size ?</i>	XYPair	Describes the length and width of the pallet, in points (e.g., 3500 3500). If not specified, the size is defined by <i>@PalletType</i> .

8.69 PalletizingParams

PalletizingParams defines the details of **Palletizing**. Details of the actual pallet used for **Palletizing** can be found in the **Pallet** Resource that is also an input of the **Palletizing** Process.

Resource Properties

Resource referenced by: —

Intent Pairing:

Input of Processes: **Palletizing**

Output of Processes: —

Table 8-150: PalletizingParams Resource

Name	Data Type	Description
<i>LayerAmount</i> ?	IntegerList	Ordered number of input components in a layer. The first number is the first layer on the bottom. If there are more layers than entries in the list, counting restarts at the first entry.
<i>MaxHeight</i> ?	double	Maximum height of a loaded pallet in points.
<i>MaxWeight</i> ?	double	Maximum weight of a loaded pallet in grams.
<i>Overhang</i> ?	XYPair	Overhang in <i>x</i> and <i>y</i> direction on each side.
<i>OverhangOffset</i> ?	XYPair	Overhang offset if overhang is not centered.
<i>Pattern</i> ?	string	Name of the palletizing pattern. Used to store a predefined pattern that defines the layers and positioning of individual component on the pallet.

8.70 PDLCreationParams

This Resource describes the details of generating the supported output PDL types used in the **PDLCreation** Process.

Resource Properties

Resource referenced by: —

Input of Processes: **PDLCreation**

Output of Processes: —

Table 8-151: PDLCreationParams Resource

Name	Data Type	Description
<i>MimeType</i>	string	This Resource identifies the MIME type associated with this output file format. For example " <i>application/pdf</i> ".
<i>FontParams</i>	element	<i>Fontparams</i> describes how fonts SHALL be handled when creating PDL.
<i>PDFCreationDetails</i> ?	element	PDF specific Resource Element for the output. It SHALL NOT be specified unless @ <i>MimeType</i> = " <i>application/pdf</i> ".
<i>PSCreationDetails</i> ?	element	Postscript specific Resource Element for the output. SHALL NOT be specified unless @ <i>MimeType</i> = " <i>application/postscript</i> ".

8.70.1 Element: FontParams

This element describes how fonts are handled when converting PostScript or other PDL files to PDF.

Table 8-152: FontParams Element

Name	Data Type	Description
<i>AlwaysEmbed</i> ?	NMTOKENS	One or more names of fonts that are always to be embedded in the PDF file. Each name SHALL be the PostScript language name of the font. An entry that occurs in both the <i>@AlwaysEmbed</i> and <i>@NeverEmbed</i> lists constitutes an error.
<i>EmbedAllFonts</i> ?	boolean	If "true", specifies that all fonts, except those in the <i>@NeverEmbed</i> list, are to be embedded in the PDF file.
<i>MaxSubsetPct</i> ?	integer	If specified, fonts SHALL be subsetted if the percentage of glyphs used is below the value of <i>MaxSubsetPct</i> in percent.
<i>NeverEmbed</i> ?	NMTOKENS	One or more names of fonts that are never to be embedded in the PDF file. Each name SHALL be the PostScript language name of the font. An entry that occurs in both the <i>@AlwaysEmbed</i> and <i>@NeverEmbed</i> lists constitutes an error.

8.70.2 Element: PSCreationDetails

This Parameter specifies a set of configurable options that can be used by Processes that generate PostScript files.

Some descriptions below mention Attributes or structures in specific source formats, such as PDF. Appropriate equivalent actions should be taken when converting from other source formats that have equivalent Attributes or structures. A small number of parameters apply only to PDF sources.

Font controls are applied in the following order:

- 1 *@IncludeBaseFonts*
- 2 *@IncludeEmbeddedFonts*
- 3 *@IncludeType1Fonts*
- 4 *@IncludeType3Fonts*
- 5 *@IncludeTrueTypeFonts*
- 6 *@IncludeCIDFonts*

For example, an embedded Type-1 font follows the rule for embedded fonts, not the rule for Type-1 fonts. In other words, if *@IncludeEmbeddedFonts* is "true", and *@IncludeType1Fonts* is "false", embedded Type-1 fonts would be included in the PostScript stream.

Table 8-153: PSCreationDetails Element (Sheet 1 of 3)

Name	Data Type	Description
<i>BinaryOK</i> ?	boolean	If "true", binary data are to be included in the PostScript stream.
<i>BoundingBox</i> ?	rectangle	It is used for BoundingBox DSC comment in <i>@CenterCropBox</i> calculations and for PostScript's set-pageDevice .
<i>CenterCropBox</i> ?	boolean	If "true", the CropBox from the source document is centered on the page when the CropBox is smaller than Media-Box .
<i>GeneratePageStreams</i> ?	boolean	If "true", the Process emits individual streams of data for each page in the RunList .
<i>IgnoreAnnotForms</i> ?	boolean	If "true", ignores annotations that contain a PDF XObject form. (PDF source only).

Table 8-153: PSCreationDetails Element (Sheet 2 of 3)

Name	Data Type	Description
<i>IgnoreBG</i> ?	boolean	Ignores the BG , BG2 parameters in the PDF ExtGState dictionary, and the operand of any calls to the PostScript setblack-generation operator.
<i>IgnoreColorSeps</i> ?	boolean	If "true", ignores images for Level-I separations.
<i>IgnoreDSC</i> ?	boolean	If "true", ignores DSC (Document Structuring Conventions).
<i>IgnoreExternStreamRef</i> ?	boolean	If a PDF image Resource uses an external stream and @ <i>IgnoreExternStreamRef</i> = "true", ignores code that points to the external file. (PDF source only).
<i>IgnoreHalftones</i> ?	boolean	If "true", ignores any halftone screening in the source file.
<i>IgnoreOverprint</i> ?	boolean	Ignores OP parameters in a source PDF ExtGState dictionary setoverprint in a source PostScript file, etc.
<i>IgnorePageRotation</i> ?	boolean	If "true", ignores a "concatenation" provided at the beginning of each page that orients the page so that it is properly rotated. Used when emitting EPS.
<i>IgnoreRawData</i> ?	boolean	If "true", no unnecessary filters are to be added when emitting image data.
<i>IgnoreSeparableImagesOnly</i> ?	boolean	If "true", and if emitting EPS, ignores only CMYK and gray images.
<i>IgnoreShowPage</i> ?	boolean	If "true", ignores save-and-restore showpage in PostScript files
<i>IgnoreTransfers</i> ?	boolean	Ignores TR , TR2 parameters in a source PDF ExtGState dictionary, settransfer and setcolortransfer in a source PostScript file, etc.
<i>IgnoreTTFFontsFirst</i> ?	boolean	If "true", ignores TrueType fonts before any other fonts.
<i>IgnoreUCR</i> ?	boolean	Ignores UCR , UCR2 parameters in a source PDF ExtGState dictionary, setundercolorremoval in a source PostScript file, etc.
<i>IncludeBaseFonts</i> ?	enumeration	Determines when to embed the base fonts. The base fonts are " <i>Symbol</i> " and the plain, bold, italic and bold-italic faces of " <i>Courier</i> ", " <i>Times</i> ", and " <i>Helvetica</i> ". Allowed values are: <i>IncludeNever</i> <i>IncludeOncePerDoc</i> <i>IncludeOncePerPage</i>
<i>IncludeCIDFonts</i> ?	enumeration	Determines when to embed CID fonts. Allowed values are from: @ <i>IncludeBaseFonts</i> .
<i>IncludeEmbeddedFonts</i> ?	enumeration	Determines when to embed fonts in the document that are embedded in the source file. This Attribute overrides the @ <i>IncludeType1Fonts</i> , @ <i>IncludeTrueTypeFonts</i> and @ <i>IncludeCIDFonts</i> Attributes. Allowed values are from: @ <i>IncludeBaseFonts</i> .

Table 8-153: PSCreationDetails Element (Sheet 3 of 3)

Name	Data Type	Description
<i>IncludeOtherResources</i> ?	enumeration	Determines when to include all other types of Resources in the file. Allowed values are from: @ <i>IncludeBaseFonts</i> .
<i>IncludeProcSets</i> ?	enumeration	Determines when to include ProcSets in the file. Allowed values are from: @ <i>IncludeBaseFonts</i> .
<i>IncludeTrueTypeFonts</i> ?	enumeration	Determines when to embed TrueType fonts. Allowed values are from: @ <i>IncludeBaseFonts</i> .
<i>IncludeType1Fonts</i> ?	enumeration	Determines when to embed Type-1 fonts. Allowed values are from: @ <i>IncludeBaseFonts</i> .
<i>IncludeType3Fonts</i> ?	enumeration	Determines when to embed Type-3 fonts. It is included here to complete the precedence hierarchy. It has only one value. Allowed values are: "IncludeOncePerPage"
<i>OutputType</i> ?	enumeration	Describes the kind of output to be generated. Allowed values are: PostScript EPS
<i>PSLevel</i> ?	integer	Number that indicates the PostScript level. Values include "1", "2" or "3".
<i>Scale</i> ?	double	Number that indicates the wide-scale factor of documents. Full size = "100".
<i>SetPageSize</i> ?	boolean	(PostScript Levels 2 and 3 only) If "true", sets page size on each page automatically. For PDF source, use MediaBox for outputting PostScript files and CropBox for EPS.
<i>SetupProcsets</i> ?	boolean	If "true", indicates that if ProcSets are included, the init/term code is also included.
<i>ShrinkToFit</i> ?	boolean	If "true", the page is scaled to fit the printer page size. This field overrides scale
<i>SuppressCenter</i> ?	boolean	If "true", suppresses automatic centering of page contents whose crop box is smaller than the page size.
<i>SuppressRotate</i> ?	boolean	If "true", suppresses automatic rotation of pages when their dimensions are better suited to landscape orientation. More specifically, the application that generates the PostScript compares the dimensions of the page. If the width is greater than the height, then pages are not rotated if @ <i>SuppressRotate</i> = "true". On the other hand, if @ <i>SuppressRotate</i> = "false", the orientation of each source page (e.g., as set by the PDF Rotate key) is honored, regardless of the dimensions of the pages (as defined by the MediaBox Attribute).
<i>TTasT42</i> ?	boolean	If including TrueType fonts, converts to Type-42 instead of Type-1 fonts when @ <i>TTasT42</i> = "true".
<i>UseFontAliasNames</i> ?	boolean	If "true", font alias names are used when printing with system fonts.

8.70.3 Element: PDFCreationDetails

This Element contains the parameters that control the conversion any PDL to PDF documents.

Some descriptions below mention Attributes or structures in specific source formats, such as PostScript. Appropriate equivalent actions should be taken when converting from other source formats that have equivalent Attributes or structures. A small number of s apply only to PostScript sources.

Table 8-154: PDFCreationDetails Element (Sheet 1 of 2)

Name	Data Type	Description
<i>AllowJBIG2Globals</i> ?	boolean	This Resource allows JBIG2 compressed images to share a single global dictionary in the resulting PDF file instead of a dictionary per image.
<i>ASCII85EncodePages</i> ?	boolean	If "true", binary streams (e.g., page contents streams, sampled images, and embedded fonts) are ASCII85-encoded, resulting in a PDF file that is almost pure ASCII. If "false", they are not, resulting in a PDF file that can contain substantial amounts of binary data.
<i>AutoRotatePages</i> ?	enumeration	Allows the Device to try to orient pages based on the predominant text orientation. If the source is PostScript, this Attribute is only used if the file does not contain “%%ViewingOrientation”, “%%PageOrientation” or “%%Orientation” DSC comments. If the file does contain such DSC comments, it honors them. “%%ViewingOrientation” takes precedence over others, then “%%PageOrientation”, then “%%Orientation”.
		Allowed values are: <i>None</i> – Turns @AutoRotatePages off. <i>All</i> – Takes the predominant text orientation across all pages and rotates all pages the same way. <i>PageByPage</i> – Does the rotation on a page-by-page basis, rotating each page individually. Useful for documents that use both portrait and landscape orientations.
<i>Binding</i> ?	enumeration	Determines how the printed pages would be bound.
		Allowed values are: <i>Left</i> – for left binding. <i>Right</i> – for right binding.
<i>CompressPages</i> ?	boolean	Enables compression of pages and other content streams like forms, patterns and Type 3 fonts. If "true", use Flate compression.
<i>DefaultRenderingIntent</i> ?	enumeration	Selects the rendering intent for the current Job.
		Allowed values are: <i>Perceptual</i> <i>Saturation</i> <i>RelativeColorimetric</i> <i>AbsoluteColorimetric</i>
		Note: See the <i>Portable Document Format Reference Manual</i> for more information on rendering intent.
<i>DetectBlend</i> ?	boolean	Enables or disables blend detection. If "true" and if @PDFVersion is 1.3 or higher, then blends will be converted to smooth shadings.

Table 8-154: PDFCreationDetails Element (Sheet 2 of 2)

Name	Data Type	Description
<i>DoThumbnails?</i>	boolean	If "true", thumbnails are created.
<i>InitialPageSize?</i>	XYPair	Defines the initial page dimensions, in points, that will be used to set MediaBox. This will be overridden by any page size Attribute found in the source document, such as the PostScript PageSize page Device parameter. The use of this Attribute is strongly encouraged when processing EPS files (%%BoundingBox comments do not override @ <i>InitialPageSize</i>).
<i>InitialResolution?</i>	XYPair	Defines the initial horizontal and vertical resolution, in dpi. This will be overridden by any resolution Attribute found in the source document, such as the PostScript HWResolution page Device parameter. The use of this Attribute is strongly encouraged when processing EPS files.
<i>Optimize?</i>	boolean	If "true", the PS-to-PDF converter optimizes the PDF file. See [PDF1.6] for more information on optimization.
<i>OverPrintMode?</i>	integer	Controls the overprint mode strategy of the Job. Set to "0" for full overprint or "1" for non-zero overprint. For more information, see [Adb-TN5044].
<i>PDFVersion?</i>	double	Specifies the version number of the PDF file produced. Values include all legal version designators (e.g., 1.2, 1.5).
<i>AdvancedParams?</i>	element	Advanced parameters which control how certain features of PDF are handled.
<i>PDFXParams?</i>	element	PDF/X parameters.
<i>ThinPDFParams?</i>	element	Parameters that control the optional content or form of PDF files that will be created.

8.70.4 Element: AdvancedParams

Table 8-155: AdvancedParams Element (Sheet 1 of 3)

Name	Data Type	Description
<i>AllowPSXObjects?</i>	boolean	If "true", allows PostScript XObjects.
<i>AllowTransparency?</i>	boolean	If "true", allows transparency in the PDF.
<i>AutoPositionEPSInfo?</i>	boolean	If "true", the Process automatically resizes and centers information from EPS source files on the page. (EPS source only)
<i>EmbedJobOptions?</i>	boolean	If "true", the PDF settings used to create the PDF are embedded in the PDF.
<i>EmitDSCWarnings?</i>	boolean	If "true", warning messages about questionable or incorrect DSC comments appear during the processing of the source PostScript file. (PostScript source only)
<i>LockDistillerParams?</i>	boolean	If "true", any PDFCreationDetails settings configured by the source content (e.g., with setdistillerparams in a PostScript source document) are ignored. If "false", each parameter defined in the source document overrides that set in the XJDF .

Table 8-155: AdvancedParams Element (Sheet 2 of 3)

Name	Data Type	Description
<code>ParseDSCCommentForDocInfo ?</code>	boolean	If "true", the Process parses the DSC comments in a PostScript source file and extracts the document information. This information is recorded in the Info dictionary of the PDF file.
<code>ParseDSCComments ?</code>	boolean	If "true", the Process parses the DSC comments in a PostScript source document for any information that might be helpful for converting the file or for information that is to be stored in the PDF file. If "false", the Process treats the DSC comments as pure PS comments and ignores them. (PostScript source only)
<code>PassThroughJPEGImages ?</code>	boolean	If "true", JPEG images are passed through without recompressing them.
<code>PreserveCopyPage ?</code>	boolean	If "true", the copypage operator of PostScript Level 2 is maintained. If "false", the PostScript Level 3 definition of copypage operator is used. In PostScript Levels 1 and 2, the copypage operator transmits the page contents to the current output Device (similar to showpage). However, copypage does not perform many of the re-initializations that showpage does. Many PostScript Level 1 and 2 programs used the copypage operator to perform such operations as printing multiple copies and implementing forms. These programs produce incorrect results when interpreted using the Level 3 copypage semantics. This Attribute provides a mechanism to retain Level 2 compatibility for this operator. (PostScript source only)
<code>PreserveEPSInfo ?</code>	boolean	If "true", preserves the EPS information in a PostScript source file and stores it in the resulting PDF file. (PostScript source only)
<code>PreserveHalftoneInfo ?</code>	boolean	If "true", passes halftone screen information (frequency, angle and spot function) into the PDF file. If "false", halftone information is not passed in.
<code>PreserveOPIComments ?</code>	boolean	If "true", encapsulates Open Prepress Interface (OPI) low resolution images as a form and preserves information for locating the high resolution images.
<code>PreserveOverprintSettings ?</code>	boolean	If "true", passes the value of the setoverprint operator through to the PDF file. Otherwise, overprint is ignored.
<code>TransferFunctionInfo ?</code>	enumeration	Determines how transfer functions are handled. Allowed values are: <i>Preserve</i> – Transfer functions are passed into the PDF file. <i>Remove</i> – Transfer functions are ignored. They are neither applied to the color values nor passed into the PDF file. <i>Apply</i> – Transfer functions are used to modify the data that are written to the PDF file, instead of writing the transfer function itself to the file.

Table 8-155: AdvancedParams Element (Sheet 3 of 3)

Name	Data Type	Description
UCRAndBGInfo?	enumeration	Determines whether the under-color removal and black-generation parameters from the source document (e.g., the arguments to the PostScript commands setundercolorremoval and setblackgeneration) are passed into the PDF file. Allowed values are: <i>Preserve</i> – The arguments are passed into the PDF file. <i>Remove</i> – The arguments are ignored.
UsePrologue?	boolean	If "true", the Process SHALL append a PostScript prologue file before beginning of the Job and append a PostScript epilog file after the end the Job. Such files are used to control the PostScript environment for the conversion Process. The expected location and allowable contents for these files is defined by the Process implementation. (PostScript source only)

8.70.5 Element: PDFXParams

Parameters for generating PDF/X files. Note that TrimBox, BleedBox, output intent and the Trapped state may be provided by the use of the **pdfmark** operator in a PostScript source file.

Table 8-156: PDFXParams Element (Sheet 1 of 2)

Name	Data Type	Description
PDFXBleedBoxToTrimBoxOffset?	rectangle	If the BleedBox entry is not specified in the page object of the source document, BleedBox is set to PDF TrimBox with offsets. All numbers SHALL be greater than or equal to 0.0. PDF BleedBox will be completely outside PDF TrimBox .
PDFXCheck?	NMTOKENS	List of PDF/X versions that the output SHALL be compliant with. Values include: <i>X1a</i> – see the PDF/X-1a standard [ISO15930-1:2001]. <i>X3</i> – see the PDF/X-3 standard [ISO15930-3:2002]. <i>X4</i> – see the PDF/X-4 standard [ISO15930-7:2010]. <i>X5</i> – see the PDF/X-5 standard [ISO15930-8:2010].
PDFXCompliantPDFOnly?	boolean	If "true", produces a PDF document only if PDF/X compliance tests are passed.
PDFXNoTrimBoxError?	boolean	If "true" and both TrimBox and ArtBox entries are not specified in the page object of the source document, the condition is reported as an error.
PDFXSetBleedBoxToMediaBox?	boolean	If "true" and the BleedBox entry is not specified in the page object of the source document, BleedBox is set to MediaBox .

Table 8-156: PDFXParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>PDFXTrapped</i> ?	enumeration	If a source document does not specify a Trapped state, then the value provided here is used. The value " <i>Unknown</i> " is to be used for workflows requiring 1) that the document specify a Trapped state and 2) that compliance checking fail if Trapped is not present in the document. Allowed values are: <i>Unknown</i> <i>false</i> <i>true</i>
<i>PDFXTrimBoxToMediaBoxOffset</i> ?	rectangle	If both the TrimBox and ArtBox entries are not specified in the page object of the source document, TrimBox is set to MediaBox with offsets. All numbers SHALL be greater than or equal to 0.0. The TrimBox will be completely inside MediaBox .
<i>FileSpec</i> (ReferenceOutputProfile) ?	element	An ICC profile.

8.70.6 Element: ThinPDFParams

Table 8-157: ThinPDFParams Element

Name	Data Type	Description
<i>FilePerPage</i> ?	boolean	If "true", the Process generates 1 PDF file per page.
<i>SidelineEPS</i> ?	boolean	If "true", embedded EPS files in PostScript source documents are not converted but are stored in external files in the same location as the PDF itself. (PostScript source only)
<i>SidelineFonts</i> ?	boolean	If "true", font data are stored in external files during PDF generation.
<i>SidelineImages</i> ?	boolean	If "true", image data are stored in an external stream during the PDF Generation phase. This prevents large amounts of image data from having to be passed through all phases of the code generation Process.

8.71 PerforatingParams

PerforatingParams define the parameters for perforating a Sheet.

Resource Properties

Resource referenced by:

—

Intent Pairing:

FoldingIntent

Input of Processes:

Perforating

Output of Processes:

—

Table 8-158: PerforatingParams Resource

Name	Data Type	Description
Perforate *	element	Defines one or more <i>Perforate</i> lines.

8.72 PreflightParams

The **PreflightParams** Resource specifies the tests for the **Preflight** Process to run. Currently, only references to proprietary preflight profiles are supported.

Resource Properties

Resource referenced by: **PreflightReport**

Input of Processes: **Preflight**

Output of Processes: —

Table 8-159: PreflightParams Resource

Name	Data Type	Description
FileSpec ?	element	File that describes the preflight actions in a computer-readable, non-XJDF format.
PreflightTest *	element	Description of an individual test.

8.72.1 Element: PreflightTest

PreflightTest describes an individual preflight test.

Table 8-160: PreflightTest Element (Sheet 1 of 2)

Name	Data Type	Description
Action ?	enumeration	Action that should be taken whenever this test fails. Allowed values are: <i>Abort</i> – Abort preflight and stop further checks. <i>Continue</i> – Log the condition and continue checking. <i>Repair</i> – Repair the condition in a system specified manner, Log the condition and continue checking.
DescriptiveName ?	element	Human readable preflight test description.
Severity ?	enumeration	Severity of a failure of the test in order of most severe to least severe are: Allowed values are: <i>Fatal</i> <i>Error</i> <i>Warning</i> <i>Information</i> Note: if not specified, an implementation MAY generate <i>PreflightCheck/@Severity</i> based on the details of test.

Table 8-160: PreflightTest Element (Sheet 2 of 2)

Name	Data Type	Description
<i>TestClass</i> ?	NMTOKEN	General area of the preflight test. Values include: <i>Resolution</i> – Tests that check low resolution graphics <i>Font</i> – Tests that check font integrity. <i>Colorspace</i> – Tests that check color space violations. <i>PageFormat</i> – Tests that detect invalid page sizes or page boxes <i>FileFormat</i> – Tests that detect incorrect file format compliancy.
<i>TestID</i> ?	NMTOKEN	System dependent preflight test identifier
<i>GeneralID</i> *	element	Detailed individual s of the <i>PreflightTest</i> . The values of <i>GeneralID/@IDUsage</i> and <i>GeneralID/@IDValue</i> are system dependent.

```

<PreflightParams Class="Parameter" ID="PP001" Status="Available">
  <TestPool>
    <Test ID="PT01">
      <BooleanEvaluation ValueList="true">
        <BasicPreflightTest Name="InsideBox">
          <PreflightArgument>
            <BoxArgument Box="TrimBox" Overlap="true"/>
          </PreflightArgument>
        </BasicPreflightTest>
      </BooleanEvaluation>
    </Test>
  </TestPool>
  <ActionPool/>
</PreflightParams>

```

8.73 PreflightReport

The **PreflightReport** Resource describes the results of the preflight tests specified in **PreflightParams**.

Resource Properties

Resource referenced by: —

Input of Processes: *Preflight*

Output of Processes: *Preflight*

Table 8-161: PreflightReport Resource

Name	Data Type	Description
<i>ErrorCount</i> ?	integer	The count of errors that were encountered while preflighting the Job.
<i>WarningCount</i> ?	integer	The count of warnings that were encountered while preflighting the Job.
<i>FileSpec</i> ?	element	References a human readable preflight report
<i>PreflightCheck</i> *	element	List of individual preflight results.

8.73.1 Element: PreflightCheck

PreflightCheck describes an individual preflight occurrence or set of similar occurrences. These occurrences MAY be distributed over multiple pages of the document that was preflighted.

Table 8-162: PreflightCheck Element

Name	Data Type	Description
Action?	enumeration	Action that has been taken. Allowed values are from PreflightTest/@Action
Count?	integer	The total number of occurrences of this PreflightCheck.
Pages?	IntegerList	A 0-based index of Pages in the document where one or more errors of the type specified by this PreflightCheck occurred.
Severity?	enumeration	Severity of the PreflightCheck. If the <i>Preflight</i> process is described in detail with PreflightTest elements that include PreflightTest/@Severity, then this SHALL be a copy of the appropriate PreflightTest/@Severity. Allowed value are from: PreflightTest/@Severity.
TestID?	NMTOKEN	System dependent error identifier. If the <i>Preflight</i> process is described in detail with PreflightTest elements, then this SHALL be a copy of the appropriate PreflightTest/@TestID.
TestClass?	NMTOKEN	General area of the preflight check. If the <i>Preflight</i> process is described in detail with PreflightTest elements, then this SHALL be a copy of the appropriate PreflightTest/@TestClass. Allowed values are from PreflightTest/@TestClass
Comment?	element	Human readable preflight check description.
GeneralID *	element	Detailed individual parameters of the PreflightTest. The values of GeneralID/@IDUsage and GeneralID/@IDValue are system dependent.

```

<PreflightReport Class="Parameter" ID="PP001" Status="Available"
    ErrorCount="0" WarningCount ="0" >
    <PRItem Occurrences="4" ActionRef="A001">
        <PRGroup Occurrences="1">
            <PRGroupOccurrence PageNumber="1"/>
            <PROccurrence Occurrences="20">
                <PRGroup Occurrences="5">
                    <PRGroupOccurrence/>
                    <PROccurrence TextSize="12"/>
                </PRGroup>
                <PRGroup Occurrences="15">
                    <PRGroupOccurrence/>
                    <PROccurrence EffectiveResolution="300 300"/>
                </PRGroup>
            </PROccurrence>
        </PRGroup>
        <PRGroup Occurrences="1">
            <PRGroupOccurrence PageNumber="4"/>
            <PROccurrence Occurrences="20">
                <PRGroup Occurrences="7">
                    <PRGroupOccurrence/>
                    <PROccurrence NumberOfPathPoints="4"/>
                </PRGroup>
            </PROccurrence>
        </PRGroup>
    </PRItem>

```

```

</PRGroup>
<PRGroup Occurrences="13">
    <PRGroupOccurrence/>
    <PROccurrence EffectiveResolution="300 300"/>
</PRGroup>
</PROccurrence>
</PRGroup>
<PRGroup Occurrences="1">
    <PRGroupOccurrence PageNumber="8"/>
</PRGroup>
<PRGroup Occurrences="1">
    <PRGroupOccurrence PageNumber="12"/>
</PRGroup>
</PRIitem>
<PreflightParams>
    <TestPool>
        <Test ID="T001">
            <BooleanEvaluation ValueList="true"/>
        </Test>
    </TestPool>
    <ActionPool>
        <Action ID="A001" TestRef="T001"/>
    </ActionPool>
</PreflightParams>
<PreflightReportRulePool/>
<RunList/>
</PreflightReport>

```

8.74 Preview

The preview of the content of a surface. It can be used for the calculation of the ink coverage (@*PreviewUsage* = "Separation") or as a preview of what is currently processed in a Device (@*PreviewUsage* = "Viewable" or @*PreviewUsage* = "Thumbnail"). When the preview is of @*PreviewUsage* = "Separation" or @*PreviewUsage* = "SeparationRaw", a gray value of "0" represents full ink, while a value of "255" represents no ink (for more information, see DeviceGray color model chapter 4.8.2 of the *PostScript Language Reference Manual*) [PS].

Resource Properties

Resource referenced by: JDF, Product, ResourceSet, Resource, QueueEntry

Input of Processes: *InkZoneCalculation, PreviewGeneration*

Output of Processes: *PreviewGeneration*

Table 8-163: Preview Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>Compensation ?</i>	enumeration	<p>Compensation of the image to reflect the application of transfer curves to the image.</p> <p>Allowed values are:</p> <p><i>None</i> – No compensation.</p> <p><i>Film</i> – Compensated until film exposure.</p> <p><i>Plate</i> – Compensated until plate exposure.</p> <p><i>Press</i> – Compensated until press.</p>

Table 8-163: Preview Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>CTM</i> ?	matrix	Orientation of the Preview with respect to the Layout coordinate system. CTM is applied after any transformation defined within the referenced image file (e.g., the transformation defined in the CIP3PreviewImageMatrix of a PPF file). In case of PPF, @ <i>CTM</i> is applied to the native Postscript coordinate system of the preview. In case of PNG, the origin of the object is defined as the lower left corner of the image.
<i>PreviewFileType</i> ?	NMTOKEN	<p>The file type of the preview.</p> <p>Values include:</p> <p><i>PNG</i> – The Portable Network Graphics format.</p> <p><i>CIP3Multiple</i> – The format as defined in the CIP3 PPF specification. One or more previews per CIP3 file are supported.</p> <p><i>CIP3Single</i> – The format as defined in the CIP3 PPF specification. Only one preview per CIP3 file is supported.</p> <p>Values include also: any MIME media type. See Appendix H, “MimeType and MimeVersion Attributes” on page 1197.</p> <p>Note: The CIP3 formats were added in JDF 1.2 only for backwards compatibility since many systems only support CIP3 format. The CIP3 formats SHALL NOT be used except in Preview Resources that are used as Input Resources to InkZoneCalculation.</p>
FileSpec	element	FileSpec SHALL identify the preview file (e.g., the PNG image or CIP3 PPF file that represents this Preview).

8.75 PreviewGenerationParams

Parameters specifying the size and the type of the preview.

Resource Properties

Resource referenced by: —

Input of Processes: *PreviewGeneration*

Output of Processes: —

Table 8-164: PreviewGenerationParams Resource

Name	Data Type	Description
<i>AspectRatio</i> ?	enumeration	<p>Policy that defines how to define the preview size if the aspect ratio of the source and preview are different. Note that <i>@AspectRatio</i> only has an effect if <i>@Size</i> is specified.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>CenterMax</i> – Keep the aspect ratio and preview <i>@Size</i>, and center the image so that the preview has missing pixels at both sides of the larger dimension. <i>CenterMin</i> – Keep the aspect ratio and preview <i>@Size</i>, and center the image so that the preview has blank pixels at both sides of the smaller dimension. <i>Crop</i> – Keep the aspect ratio, and modify the preview size so that the image fits into a bounding rectangle defined by <i>@Size</i>. <i>Expand</i> – Keep the aspect ratio, and modify the preview size so that the smaller image dimension is defined by <i>@Size</i>. <i>Ignore</i> – Fill the preview completely, keeping <i>@Size</i>, even if this requires modifying the aspect ratio.
<i>Compensation</i> ?	enumeration	<p>Compensation of the image to reflect the application of transfer curves to the image.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>None</i> – No compensation. <i>Film</i> – Compensated until film exposure. <i>Plate</i> – Compensated until plate exposure. <i>Press</i> – Compensated until press.
<i>PreviewFileType</i> ?	enumeration	<p>The file type of the preview to be generated.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>PNG</i> – The Portable Network Graphics format. <i>CIP3Multiple</i> – The format as defined in the CIP3 PPF specification. One or more previews per CIP3 file are supported. <i>CIP3Single</i> – The format as defined in the CIP3 PPF specification. Only one preview per CIP3 file is supported. <p>Note: The CIP3 formats were added in JDF 1.2 only for backwards compatibility since many systems only support CIP3 format. The CIP3 formats SHALL NOT be used except in Preview Resources that are used as Input Resources to InkZoneCalculation.</p>
<i>Resolution</i> ?	XYPair	Resolution of the preview, in dpi. If <i>@PreviewUsage</i> = "Separation", the default is "50.8 50.8".
<i>Size</i> ?	XYPair	Size of the preview, in pixels. If this Attribute is present, the <i>@Resolution</i> Attribute evaluated according to the policy defined in <i>@AspectRatio</i> . If <i>@Size</i> is not specified, it SHALL be calculated using the <i>@Resolution</i> Attribute and the input image size.

8.76 PrintCondition

PrintCondition is a Resource used to control the use of colorants when printing pages on a specific media. The Attributes and Elements of the **PrintCondition** Resource describe the aim values for a given printing Process.

Resource Properties

Resource referenced by:	—
Input of Processes:	<i>ConventionalPrinting, DigitalPrinting</i>
Output of Processes:	—

Table 8-165: PrintCondition Resource

Name	Data Type	Description
<i>AimCurve</i> ?	Transfer-Function	Describes the desired tone-value increase function. If not specified, it defaults to the media and printing machine-specific values
<i>Density</i> ?	double	Density value of colorant (100% tint). Whereas Color / <i>@NeutralDensity</i> describes measurements of inks on substrate with wide-band filter functions, @Density is derived from measurements of inks on substrate with special small band filter functions according to ANSI and DIN. If not specified, it defaults to the value of Color // @Density .
<i>Name</i>	string	Name of the PrintCondition . Used to reference a PrintCondition from a Color / Element.
ColorMeasurementConditions ?	element	Describes measurement conditions for color measurement and density measurement. If not specified, it defaults to the value of Color // ColorMeasurementConditions
Device ?	element	Specifies the Device or Device group that this PrintCondition applies to.
FileSpec (<i>TargetProfile</i>) ?	element	A FileSpec Resource pointing to an ICC profile that defines the target output Device in case the object that uses the Color has been color space converted to a Device color space. If not specified, it defaults to the value of Color // FileSpec (<i>TargetProfile</i>).

Example 8-27: PrintCondition

TBD 2.x Example.

```
<ColorMeasurementConditions Class="Parameter" ID="MyColorMeasCond"
    Status="Available"/>
<PrintCondition Name="Standard" Class="Parameter" ID="PC"
    PartIDKeys="Side Separation" Status="Available">
    <ColorMeasurementConditionsRef rRef="MyColorMeasCond"/>
    <PrintCondition Side="Front">
        <PrintCondition AimCurve="0.0 0.0 0.5 0.66 1.0 1.0" Density="1.8"
            Separation="Black"/>
        <PrintCondition AimCurve="0.0 0.0 0.5 0.63 1.0 1.0" Density="1.4"
            Separation="Cyan"/>
    </PrintCondition>
</PrintCondition>
```

8.77 ProductionPath

ProductionPath describes the individual paper path through the different modules of a Web-Press Device, in order to produce a particular product.

Resource Properties

Resource referenced by: —

Input of Processes: *WebInlineFinishing*

Output of Processes: —

Table 8-166: ProductionPath Resource

Name	Data Type	Description
<i>ProductionPathID</i> ?	NMTOKEN	Identification of the entire production path. The <i>@ProductionPathID</i> SHALL be unique within the machine.

Example 8-28: ProductionPath: on Path Level:

This example and the next illustrate the different Web path description levels:

TBD 2.x Example.

```
<ProductionPath Class="Parameter" ID="F1" Status="Available"
    ProductionPathID="ID_2webproduction_64pages"/>

<ProductionPath Class="Parameter" ID="F1" Status="Available"
    PartIDKeys="WebName">
    <ProductionPath WebName="1">
        <PrintingUnitWebPath ProductionPathID="ID_PrintingUnitWebPath"/>
        <FolderSuperstructureWebPath ProductionPathID="abcd"/>
        <PostPressComponentPath ProductionPathID="xyz"/>
    </ProductionPath>
</ProductionPath>
```

8.78 QualityControlParams

This set of parameters identifies how the *QualityControl* Process is to operate. The **QualityControlParams** defines the generic set of parameters for the quality control Process. The specific measurement conditions are defined in specialized Subelements such as *BindingQualityParams*.

Resource Properties

Resource referenced by: —

Input of Processes: *QualityControl*

Output of Processes: —

Table 8-167: QualityControlParams Resource

Name	Data Type	Description
<i>SampleInterval</i> ?	integer	Interval in number of samples between tests.
<i>TimeInterval</i> ?	duration	Time interval between individual tests.
<i>BindingQualityParams</i> ?	element	Specification of the binding quality measurements..

8.78.1 Element: BindingQualityParams

The set of parameters in *BindingQualityParams* identifies how the quality of the binding is verified.

Table 8-168: BindingQualityParams Element

Name	Data Type	Description
FlexValue ?	double	Flex quality parameter measured in [N/cm].
PullOutValue ?	double	Pull out quality parameter measured in [N/cm].

8.79 QualityControlResult

This set of parameters returns results of a **QualityControl** Process. The **QualityControlResult** defines the generic set of results from the quality control Process. The specific measurements are returned in specialized Subelements such as **BindingQualityParams**. Additional detailed quality control result types are anticipated in future versions of the XJDF specification.

Resource Properties

Resource referenced by:

Input of Processes:

Output of Processes: **QualityControl**

Table 8-169: QualityControlResult Resource

Name	Data Type	Description
Condition ?	NMTOKEN	Condition of the tested Component . If the Component passed the test, but the test itself destroyed the Component , the value SHALL be set to "destroyed". Values include: <i>destroyed</i>
End ?	dateTime	Date and time of the end of the measurement. If not specified, the value of @Start is applied.
Failed ?	integer	Total number of failed measurements.
Passed ?	integer	Total number of passed measurements.
Start ?	dateTime	Date and time of the start of the measurement.
FileSpec ?	element	Location of an external file that contains details of the quality control measurement.

8.80 RasterReadingParams

This set of parameters specifies the details for **RasterReading**.

Resource Properties

Resource referenced by:

Input of Processes: **RasterReading**

Output of Processes:

Table 8-170: RasterReadingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
Center ?	boolean	Indicates whether or not the finished page image is to be centered within the imageable area of the media. The @Center is ignored if FitPolicy/@SizePolicy = "ClipToMaxPage" and clipping is requested.

Table 8-170: RasterReadingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>FilmRef?</i>	IDREF	Reference to film Media . This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during RasterReading .
<i>MirrorAround?</i>	enumeration	This Attribute specifies the axis around which a raster reader is to mirror an image. Allowed values are: <i>None</i> – The default. <i>FeedDirection</i> – Image is mirrored around the feed-direction axis. <i>MediaWidth</i> – Image is mirrored around the media-width axis. <i>Both</i> – Image is mirrored around both possible axes.
<i>PaperRef?</i>	IDREF	Reference to final paper Media . This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during RasterReading .
<i>PlateRef?</i>	IDREF	Reference to plate Media . This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during RasterReading .
<i>Polarity?</i>	enumeration	The image SHALL be RIPed in the polarity specified. Note that this is a polarity change in the RIP and not a polarity change in the hardware of the output Device. Allowed values are: <i>Positive</i> <i>Negative</i>
<i>ProofPaperRef?</i>	IDREF	Reference to paper Media used for proofing. This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during RasterReading .
<i>Scaling?</i>	XYPair	A pair of positive real values that indicates the scaling factor for the page contents. Values between 0 and 1 specify that the contents are to be reduced, while values greater than 1 specify that the contents are to be expanded. This Attribute is ignored if <i>@FitToPage = "true"</i> or if <i>@Poster</i> is present and has a value other than " <i>1 1</i> ". Any scaling defined in FitPolicy SHALL be applied after the scaling defined by this Attribute.
<i>ScalingOrigin?</i>	XYPair	A pair of real values that identify the point in the unscaled page that is to become the origin of the new, scaled page image. This point is defined in the coordinate system of the unscaled page. If not specified, and scaling is requested, the <i>@ScalingOrigin</i> defaults to " <i>0 0</i> "
<i>FitPolicy?</i>	element	Allows printing even if the size of the imageable area of the media does not match the requirements of the data. This replaces the deprecated <i>@FitToPage</i> Attribute.

8.81 RegisterMark

Defines a register mark, which can be used for setting up and monitoring color registration in a printing Process. It can also be used to synchronize the Sheet position in a paper path. The position and rotation of each register mark can

be specified with the help of the following Attributes. It is important that the register marks are defined in such a way that their centers are on the point of origin of the coordinate system, as otherwise they are not positioned properly.

Resource Properties

Resource referenced by: **Layout/MarkObject**

Input of Processes: —

Output of Processes: —

Table 8-171: RegisterMark Resource

Name	Data Type	Description
<i>Center</i> ?	XYPair	Position of the center of the register mark in the coordinates of the MarkObject that contains this mark.
<i>MarkType</i> ?	NMTOKENS	Type of RegisterMark . Values include: <i>Arc</i> <i>Circle</i> <i>Cross</i>
<i>MarkUsage</i> ?	enumerations	Specifies the usage of the RegisterMark . Allowed values are: <i>Color</i> – The mark is used for separation color registration. <i>PaperPath</i> – The mark is used for paper path synchronization. <i>Tile</i> – The mark is used to mark the position of tiles in Tiling.
<i>Rotation</i> ?	double	Rotation in degrees. Positive graduation figures indicate counter-clockwise rotation; negative figures indicate clockwise rotation.
<i>Separations</i> ?	NMTOKENS	Set of separations to which the register mark is bound.

8.82 RenderingParams

This set of parameters identifies how the **Rendering** Process is to operate. Specifically, these parameters define the expected output of the **ByteMap** Resource that the **Rendering** Process creates.

Resource Properties

Resource referenced by: —

Intent Pairing: **ContentCheckIntent**

Input of Processes: **Rendering**

Output of Processes: —

Table 8-172: RenderingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>BandHeight</i> ?	integer	Height of output bands expressed in lines. For a frame Device, the band height is simply the full height of the frame.

Table 8-172: RenderingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>BandOrdering</i> ?	enumeration	Indicates whether output buffers are generated in " <i>BandMajor</i> " or " <i>ColorMajor</i> " order. Allowed values are: <i>BandMajor</i> – The position of the bands on the page is prioritized over the color. <i>ColorMajor</i> – All bands of a single color are played in order before progressing to the next plane. This is only possible with non-interleaved data.
<i>BandWidth</i> ?	integer	Width of output bands, in pixels.
<i>ColorantDepth</i> ?	integer	Number of bits per colorant. Determines whether the output is bitmaps or bytemaps.
<i>Interleaved</i> ?	boolean	If " <i>true</i> ", the resulting colorant values are interleaved and @ <i>BandOrdering</i> is ignored.
<i>MimeType</i>	string	@ <i>MimeType</i> identifies the MIME type associated with this output file format. For example " <i>application/pdf</i> ".
<i>AutomatedOverPrintParams</i> ?	element	Controls for overprint substitutions. Defaults to no automated overprint generation.
<i>ObjectResolution</i> *	element	Elements which define the resolutions to render the contents at. More than one Element MAY be used to specify different resolutions for different @ <i>SourceObjects</i> types. If no <i>ObjectResolution</i> is specified, the value is implied from the input data.
<i>TIFFFormatParams</i> ?	element	Parameters specific to conversion of rasters to TIFF files.

8.82.1 Element: TIFFFormatParams

Table 8-173: TIFFFormatParams Element (Sheet 1 of 2)

Name	Data Type	Description
<i>ByteOrder</i> ?	enumeration	Byte order of the TIFF file. Allowed values are: <i>II</i> – Low byte first. <i>MM</i> – high byte first. Note: the identifier values have been selected to match the identifier with the same purpose within the TIFF file itself.
<i>Interleaving</i> ?	integer	How the components of each pixel are stored. The values are taken from TIFF tag 284— <i>PlanarConfiguration</i> : Allowed values are: 1 – “Chunky” format, which is pixel interleaved. 2 – “Planar” format, which is strip interleaved.

Table 8-173: TIFFFormatParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>RowsPerStrip</i> ?	integer	The number of image scan lines per strip, encoded in the TIFF file as <i>RowsPerStrip</i> . This Attribute is ignored if <i>@Segmentation!</i> = "Stripped". The default, when not known, is set by the processing system with the exception that when converting from <i>ByteMap</i> to TIFF, <i>ByteMap/ @BandHeight</i> is the default.
<i>Segmentation</i> ?	enumeration	How the image data are segmented. Allowed values are: <i>SingleStrip</i> – all data are included in one segment. This is encoded in the TIFF file by setting <i>@RowsPerStrip</i> to a number equal to or larger than the number of pixel rows in the image. <i>Stripped</i> – Data are segmented into strips. <i>Tiled</i> – Data are segmented into tiles.
<i>SeparationNameTag</i> ?	integer	When color separations are stored in individual TIFF files it is often useful to mark each with the name of the colorant that it represents, but there is no universally accepted way to do this. In order to avoid the need for explicit Partitioning, the tag to be used to encode the separation name (as a string) can be entered here as the TIFF tag number. If the same TIFF tag number is also supplied as a TIFFtag Subelement, then the TIFFtag Element takes priority over <i>@SeparationNameTag</i> . The tag SHOULD only be put in the resulting TIFF files if the name of the separation is known. The default of "270" is the " <i>TIFF</i> " ImageDescription tag.
<i>TileSize</i> ?	XYPair	Two integers. The X value provides width of tiles, and the Y value provides height of tiles. This Attribute is ignored if <i>@Segmentation</i> is not "Tiled".
<i>WhiteIsZero</i> ?	boolean	When writing monochrome or grayscale files, this flag indicates whether the data is to be written as "WhiteIsZero" or "BlackIsZero."
<i>TIFFEmbeddedFile</i> *	element	Files to be embedded within the created TIFF file. These might include an ICC profile, XMP data, etc.
<i>TIFFtag</i> *	element	Specific tag values for inclusion in the TIFF file.

The number of channels SHOULD be derived from the raster data to be converted.

When the **PhotometricInterpretation** tag = 5 and the **InkSet** tag = 2, it is strongly RECOMMENDED that the **NumberOfInks** and **InkNames** tags be completed—separation names MAY be obtained from **ResourceSet[@Name="Color"]**.

Flate and JPEG compression in resulting TIFF files SHOULD use Compression = 8 and Compression = 7 respectively, as documented in [TIFFPS]. In particular, the JPEG encoding using Compression = 6, as described in [TIFF6] SHOULD NOT be used.

8.82.2 Element: TIFFtag

Table 8-174: TIFFtag Element

Name	Data Type	Description
<i>BinaryValue</i> ?	hexBinary	If the type of the tag is UNDEFINED, then <i>@BinaryValue</i> is used to encode the data
<i>IntegerValue</i> ?	IntegerList	If the type of the tag is BYTE, SHORT, LONG, SBYTE, SSHORT or SLONG, then <i>@IntegerValue</i> is used to encode that data
<i>NumberValue</i> ?	DoubleList	If the type of the tag is RATIONAL, SRATIONAL, FLOAT or DOUBLE, then <i>@NumberValue</i> is used to encode that data
<i>StringValue</i> ?	string	If the type of the tag is ASCII, then <i>@StringValue</i> is used to encode the data.
<i>TagNumber</i>	integer	Tag number of the specified tag (e.g., 270 (decimal) for ImageDescription).
<i>TagType</i>	integer	The type of the tag as defined in [TIFF6] (1 = BYTE, 2 = SHORT, etc.).

Exactly one of *@IntegerValue*, *@NumberValue*, *@StringValue* or *@BinaryValue* SHALL be present, depending on the type of the TIFF tag to be carried. TIFFtag Elements SHALL NOT be used for any tags related to the image data and its encoding (ImageWidth, Compression, etc.). TIFFtag Elements MAY include informational tags such as OPIProxy, ImageID, Copyright, DateTime, ImageDescription, etc.

8.82.3 Element: TIFFEmbeddedFile

Table 8-175: TIFFEmbeddedFile Element

Name	Data Type	Description
<i>TagNumber</i>	integer	Tag number of the specified tag (e.g., 34675 (decimal) for an ICC profile or 700 for XMP).
<i>TagType</i>	integer	The type of the tag as defined in [TIFF6]. This will usually be 1 (BYTE) or 7 (UNDEFINED).
<i>FileSpec</i>	element	Reference to the file to be embedded.

8.83 RunList

RunList Resources describe an ordered set of **FileSpec** or **ByteMap** Elements. Ordering and structure are defined using the generic Partitioning mechanisms as described in Section 3.13.2.2, “Selecting a Partition”.

RunList Resources are used whenever a set of page descriptions Elements are specified. Depending on the Process usage of a **RunList**, only certain values of *@ElementType* MAY be valid. For example, a pre-RIP **Imposition** Process requires *@ElementType* to be either “*Page*” or “*Document*”, whereas a post-RIP **Imposition** Process requires **ByteMap** Elements. The usage is detailed in the descriptions of the Processes that use the **RunList** Resource. **RunList** Resources allow structuring of multiple *Pages* into *Documents*. Multiple *Documents* that have a joint context MAY be grouped into *Sets*.

In essence, a **RunList** is a virtual document or set of documents. It allows a document to either be physically spread over multiple files, or multiple documents to be contained within a single file (e.g., PPML, PDF/VT). It retains

the same properties as the original documents (e.g., the pages of a document that is described by a **RunList** are ordered).

Note: **RunList** Elements SHOULD NOT be partitioned by *@DocCopies*, *@DocIndex*, *@DocRunIndex*, *@DocSheetIndex*, *@RunIndex*, *@SetCopies*, *@SetDocIndex*, *@SetIndex*, *@SetRunIndex* or *@SetSheetIndex* unless *@Automation* = "Dynamic".

Resource Properties

Resource referenced by:	DigitalMedia, Layout/PageCondition, Layout/SheetCondition, , PreflightReport
Input of Processes:	AssetListCreation, ColorCorrection, ColorSpaceConversion, ContoneCalibration, DigitalPrinting, ImageSetting, Imposition, Interpreting, LayoutElementProduction, LayoutShifting, PDLCreation, Preflight, PreviewGeneration, RasterReading, Rendering, Screening, Separation, ShapeDefProduction, Stripping, Tiling, Trapping
Output of Processes:	AssetListCreation, ColorCorrection, ColorSpaceConversion, ContoneCalibration, Imposition, Interpreting, LayoutElementProduction, LayoutShifting, PDLCreation, RasterReading, Rendering, Screening, Separation, Stripping, Tiling, Trapping

Table 8-176: RunList Resource (Sheet 1 of 7)

Name	Data Type	Description
<i>Automation</i> ?	enumeration	<p>Identifies dynamic and static RunList Elements. The Structure of <i>@PartIDKey</i> generation for automated imposition is defined in detail in: Section 6.4.18.3, Execution Model for Automated Imposition. This structure SHALL be retained in the RunList description.</p> <p>Allowed values are:</p> <p><i>Static</i> – The RunList is static and completely qualified.</p> <p><i>Dynamic</i> – The RunList is a template. If <i>@PipeID</i> is also present, Details are specified in XJMF Pipe messages. See Section 11.3.4.1, Dynamic Pipes.</p>
<i>ClipPath</i> ?	PDFFPath	Path that describes the outline of the RunList in the coordinate space of the RunList of <i>@ElementType</i> = "Page" that results from the LayoutElementProduction Process. The default case is that there is no clip path. <i>@ClipPath</i> , <i>@SourceClipBox</i> , <i>PlacedObject</i> / <i>@SourceClipPath</i> and <i>PlacedObject</i> / <i>@ClipBox</i> if supplied, SHALL be concatenated.
<i>ContentRefs</i> ?	IDREFS	Ordered list of IDs of Content Elements. Content elements provide Metadata related to the product to be published.

Table 8-176: RunList Resource (Sheet 2 of 7)

Name	Data Type	Description
<i>Directory</i> ?	URL	Defines a directory where the files that are associated with this RunList are to be copied to or from. If <i>@Directory</i> is specified, it SHALL be an Absolute URI [RFC3986] that implicitly also specifies a Base URI which is used to resolve any relative URL of RunList . See Appendix K, “Resolving RunList/@Directory and FileSpec/@URL URI References” on page 1255 and [FileURL] for examples.
<i>DocCopies</i> ?	integer	<p>Number of Instance Document copies that this RunList represents. Specifying <i>@DocCopies</i> is equivalent to repeating the sequence of RunList leaves between <i>@EndOfDocument</i> = "true" for a total of <i>DocCopies</i> times.</p> <p>If <i>DocCopies</i> is > 1 for an automated imposition job, the imposition engine places the equivalent <i>DocCopies</i> Attribute into the RunList (<i>Surface</i>) Resource generated by the Imposition Process. An exception is cut-and-stack imposition, where <i>DocCopies</i> is applied by the imposition engine itself, and not placed into the RunList (<i>Surface</i>).</p> <p>Note: It is illegal to specify <i>DocCopies</i> with different values of various leaves of a RunList representing the same Instance Document.</p>
<i>DocNames</i> ?	RegExp	A list of named documents in a multi-document file that supports named access to individual documents. The <i>@DocNames</i> defaults to all documents. If <i>@DocNames</i> occurs in the RunList , <i>@Docs</i> is ignored if it is also present.
<i>Docs</i> ?	RegExp	Zero-based list of document indices in a multi-document file specified by the RunList Resource.
<i>DocumentNaturalLang</i> ?	language	The natural language of the document this FileSpec refers to. If the document contains more than one language, the value is the primary language of the document.
<i>ElementType</i> ?	enumeration	<p>Describes the content type for this RunList.</p> <p>Allowed values are from: Table 8-272, “ElementType Attribute Values” on page 900.</p>
<i>EndOfDocument</i> ?	boolean	If "true", the last finished page in the RunList is the last page of an Instance Document. The precise handling of Instance Document changes is defined in the InsertSheet Resource. If the RunList references a PDL that supports internal Instance Documents, <i>@EndOfDocument</i> MAY be implied from the PDL. The implied default value of <i>@EndOfDocument</i> = "false", except for the last RunList Partition leaf, which always has an implied default value of <i>@EndOfDocument</i> = "true".

Table 8-176: RunList Resource (Sheet 3 of 7)

Name	Data Type	Description
<i>EndOfSet</i> ?	boolean	If "true", the last finished page in the RunList is the last page of a set of Instance Documents. The precise handling of Instance Document boundaries is defined in the InsertSheet Resource. If the RunList references a PDL that supports internal sets, <i>@EndOfSet</i> MAY be implied from the PDL. The implied default value of <i>@EndOfSet = "false"</i> , except for the last RunList Partition leaf, which always has an implied default value of <i>@EndOfSet = "true"</i> .
<i>FinishedPages</i> ?	integer	Number of finished page surfaces that one PDL page of this RunList refers to. This attribute SHOULD be used when cover spreads or imposed sheets that contain more than one reader page per PDL page are provided.
<i>FirstPage</i> ?	integer	First finished page in the document that is described by this RunList . This Attribute is generally used to describe preseparated files.
<i>HasBleeds</i> ?	boolean	If "true", the file has bleeds.
<i>IgnorePDCopies</i> ?	boolean	If "true", any PDL defined copy count SHALL be ignored.
<i>IgnorePDLImposition</i> ?	boolean	If "true", any PDL defined imposition definition SHALL be ignored. Examples are PDF with embedded PJTF or PPML with a PRINT_LAYOUT. If <i>@IgnorePDLImposition = "false"</i> and XJDF also defines imposition, the imposed Sheets of the PDL are treated as pages in the context of XJDF imposition. The front and back surfaces of the PDL and XJDF imposition SHOULD be matched. Note that it is strongly discouraged to specify imposition both in the PDL and XJDF, and that this might result in undesired behavior.
<i>ImageCompressionParamsRef</i> ?	IDREF	Specification of the image compression properties.
<i>IsBlank</i> ?	boolean	If "true", the RunList has no content marks and is blank.
<i>IsPage</i> ?	boolean	If "true", the individual RunList Resource defines one or more page slots (e.g., for filling PlacedObject Elements). If "false", the first parent Partitioned RunList Resource with <i>@IsPage = "true"</i> defines the page level. In general, <i>@IsPage = "false"</i> for separations of a preseparated RunList .
<i>IsPrintable</i> ?	boolean	If "true", the file is a PDL file and can be printed. Possible files types include PCL, PDF or PostScript files. Application files such as MS Word have <i>@IsPrintable = "false"</i> .
<i>IsTrapped</i> ?	boolean	If "true", the file has been trapped.

Table 8-176: RunList Resource (Sheet 4 of 7)

Name	Data Type	Description
<i>LogicalPage</i> ?	integer	The logical page number of the first finished page in a RunList . This Attribute MAY be used to retain logical page indices when a Partitioned RunList is spawned. It defaults to "1" plus the last finished page of the previous sibling RunList Partition. If the RunList Resource is the first Partition, <i>@LogicalPage</i> defaults to "0". Note that is an error to specify <i>@LogicalPage</i> to be less than the number of previously defined logical pages in the same Partition, since this defines overlapping finished pages within the RunList Partition.
<i>NPage</i> ?	integer	Total number of pages (placed object slots or RunList Elements with <i>@IsPage</i> = "true") that are defined by the RunList . If <i>@NPage</i> is not specified, it defaults to all finished pages in the Partitioned RunList Elements that make up the RunList . If the RunList describes multiple Instance Documents or Document Sets, <i>@NPage</i> refers to the total number of finished pages in all Instance Documents and sets. A RunList with <i>@NPage</i> specified always refers to <i>@NPage</i> pages, regardless of the number of pages of the referenced PDL. If <i>@NPage</i> is not specified and no content is referenced, the RunList contains exactly one page.
<i>PageCopies</i> ?	integer	Number of finished page copies that this RunList represents. Specifying <i>@PageCopies</i> is equivalent to repeating the RunList leaves representing each page for a total of <i>@PageCopies</i> times (e.g., a multiple represented by the value of <i>@PageCopies</i>). Note that pages specified by <i>@PageCopies</i> are always assumed uncollected when calculating the index in the logical RunList (e.g., <i>@PageCopies</i> = "2" would result in a logical page sequence of 0 0 1 1 2 2, etc.).
<i>PageNames</i> ?	RegExp	A list of named pages in a multi-page file that supports named access to individual finished pages. The <i>@PageNames</i> defaults to all pages. If <i>@PageNames</i> is specified, then <i>@FirstPage</i> , <i>@NPage</i> , <i>@SkipPage</i> and <i>@Pages</i> SHALL all be ignored if any is specified.
<i>PageOrder</i> ?	enumeration	Indicates the order of pages in the file containing pages. Allowed values are: <i>Ascending</i> – The first page in the file is the lowest numbered page. <i>Descending</i> – The first page in the file is the highest numbered page.

Table 8-176: RunList Resource (Sheet 5 of 7)

Name	Data Type	Description
<i>Pages</i> ?	IntegerRange	Zero-based list of indices in the documents specified by the RunList Resource and the <i>@Docs</i> , <i>@DocNames</i> , <i>@Sets</i> and <i>@SetNames</i> Attributes. The <i>@Pages</i> need not be in document order. If <i>@Pages</i> is specified, <i>@FirstPage</i> and <i>@SkipPage</i> SHALL be ignored. If none of <i>@Pages</i> , <i>@FirstPage</i> , <i>@NPage</i> , <i>@PageNames</i> or <i>@SkipPage</i> is specified, all pages referred to by the RunList are selected.
<i>RunTag</i> ?	NMOKEN	Tag of a Partition of an Asset other than the RunList which is Partitioned by <i>@RunTags</i> . The Partition matches if any of the entries in the <i>@RunTags</i> list matches <i>@RunTag</i> . Multiple entries in a RunList MAY have the same <i>@RunTag</i> . If the RunList references a PDL that supports internal labels, <i>@RunTag</i> MAY be implied from the PDL.
<i>ScreeningParamsRef</i> ?	IDREF	Specification of the screening properties.
<i>Séparations</i> ?	NMOKENS	List of used separation names.
<i>SetCopies</i> ?	integer	<p>Number of Instance Document Set copies that this RunList represents. Specifying <i>@SetCopies</i> is equivalent to repeating the sequence of RunList leaves between <i>@EndOfSet = "true"</i> for a total of <i>@SetCopies</i> times.</p> <p>If <i>@SetCopies</i> is > 1 for an automated imposition Job, the imposition engine places the equivalent <i>@SetCopies</i> Attribute into the RunList (<i>Surface</i>) Resource generated by the Imposition Process. An exception is cut-and-stack imposition, where <i>@SetCopies</i> is applied by the imposition engine itself, and not placed into the RunList (<i>Surface</i>).</p> <p>Note: it is illegal to specify <i>@SetCopies</i> with different values of various leaves of a RunList representing the same Instance Document.</p>
<i>SetLevel</i> ?	XPath	<p>Specifies the mapping for the structure of a document of type MultiSet to the structure processed by the PDL Processor. If specified, the XPath expression selects a node set from the Structured PDL's hierarchy. Each node of that node set is processed by the PDL processor as an XJDF set. If not specified, the nodes that are processed as a set by the PDL processor SHALL be defined by the PDL. If the PDL does not define which nodes represent sets, then which nodes represent sets is undefined.</p> <p>Note: An example of a PDL that can define which nodes represent sets is ISO 16612-2 (PDF/VT), where the DPartRoot/@RecordLevel can provide that mapping.</p>

Table 8-176: RunList Resource (Sheet 6 of 7)

Name	Data Type	Description
<i>SetNames</i> ?	RegExp	A list of named Document Sets in a multi-Document Set file that supports named access to individual documents. The <i>@SetNames</i> defaults to all Document Sets specified by <i>@Sets</i> . If <i>@SetNames</i> occurs in the RunList , <i>@Sets</i> is ignored if it is also present. <i>@SetNames</i> is only valid if RunList / <i>@ElementType</i> = "MultiSet".
<i>Sets</i> ?	IntegerRange	Zero-based list of Document Set indices in a multi-Document Sets file specified by the RunList Resource. If not present, all Document Sets are selected. <i>@Sets</i> is only valid if RunList / <i>@ElementType</i> = "MultiSet".
<i>SheetSides</i> ?	enumeration	Specifies the binding of surfaces referenced by this RunList to sheets. SHALL only be specified in RunList (<i>Surface</i>). Allowed values are: <i>Front</i> – all surfaces referenced from a RunList leaf Partition describe one or more front sides of successive sheets, with implicit back blank sides. <i>Back</i> – all surfaces referenced from a RunList leaf Partition describe one or more back sides of successive sheets, with implicit front blank sides. <i>FrontBack</i> – all surfaces referenced from a RunList leaf Partition describe a succession of sheets, where for each sheet a front is followed by a back surface. <i>BackFront</i> – all surfaces referenced from a RunList leaf Partition describe a succession of sheets , where for each sheet a back is followed by a front surface.
<i>SkipPage</i> ?	integer	Used when the RunList comprises every Nth page of the file. <i>@SkipPage</i> indicates the number of finished pages to be skipped between each of the pages that comprise the RunList Resource. This is generally used to describe preseparated files, or to select only even or odd pages. Note that <i>@SkipPage</i> is, therefore, 3 (4 Separations -> skip 3) in a CMYK separated file.
<i>Sorted</i> ?	boolean	Specifies whether the Elements in the RunList are sorted in the document reader order.
<i>SourceBleedBox</i> ?	rectangle	A rectangle that describes the bleed area of the element to be included. This rectangle is expressed in the source coordinate system of the object.
<i>SourceClipBox</i> ?	rectangle	A rectangle that defines the region of the element to be included. This rectangle is expressed in the source coordinate system of the object.
<i>SourceMediaBox</i> ?	rectangle	The MediaBox of the RunList .
<i>SourceTrimBox</i> ?	rectangle	A rectangle that describes the intended trimmed size of the element to be included. This rectangle is expressed in the source coordinate system of the object.

Table 8-176: RunList Resource (Sheet 7 of 7)

Name	Data Type	Description
ByteMap ?	element	Describes the page or stream of pages. At most one of ByteMap or FileSpecRef SHALL be specified. If none of ByteMap or FileSpecRef are specified, the RunList specifies empty content.
FileSpec ?	element	URL plus metadata about the physical characteristics of a file representing the RunList . If not present, then only metadata is known but not the content file.
FileSpec (Font Image PDL) *	element	Reference to dependent references such as fonts, external images, etc. FileSpec/@ResourceUsage SHOULD be one of the following values: <i>Font</i> – The file references a font. <i>Image</i> – The file references Image data. <i>PDL</i> – The file references a page definition such as a PDF.
InsertSheet *	element	Describes how Sheets and Surfaces are to be completed and OPTIONAL media which MAY be inserted at the beginning or end of this RunList Resource.
MetadataMap *	element	Describes the mapping of Metadata in a RunList to @PartIDKeys .

8.83.1 Reordering the Processing of Resources

Resource Partitioning may also be used to reorder the processing order of content described by a **RunList**. This is done by using the **RunList/@IgnoreContext** Attribute, which specifies which Part Element Partition Keys' Job context should be ignored during processing. For more information and an example of this, see **RunList/@IgnoreContext** in Section 8.134, “RunList” on page 887 and following the **RunList** table, see Example 10-5, “RunList/MetadataMap” on page 1106.

Example 8-29: Marks and Reordering of Content using RunList/@IgnoreContext

Assume that a VDP job consists of sets where each set contains a Cover Letter, Brochure, and Postcard document types. Production needs all of each document type for all sets printed first, and the imposition includes dynamic marks where some of the marking uses **@SheetIndex**. The **RunListLink** parameterizes the processing such that all Cover Letter sheets for all sets are processed first, followed by the Brochure sheets for all sets, and finally, the Postcard sheets for all sets. The **RunList** then specifies **@IgnoreContext = "SheetIndex"**, which forces the **@SheetIndex** to be calculated in the order in which sheets are produced by the processing of the reordered “virtual” **RunList**.

TBD 2.x Example.

```
<ResourcePool>
  <RunList Class="Parameter" ID="MyVDPRunList" Status="Available"
    PartIDKeys="DocTags" IgnoreContext="SheetIndex" >
    <!-- additional attributes and elements -->
    <RunList DocTags="CoverLetter"/>
    <RunList DocTags="Brochure"/>
    <RunList DocTags="Postcard"/>
  </RunList>
</ResourcePool>
<ResourceLinkPool>
  <RunListLink Usage="Input" rRef="MyVDPRunList" >
    <Part DocTags="CoverLetter"/>
    <Part DocTags="Brochure"/>
```

```

<Part DocTags="Postcard"/>
</RunListLink>
</ResourceLinkPool>

```

To enable later reprinting of part of the **RunList**, the **RunList** then might also specify a **MetadataMap** Element that extracts the value of a RecordNumber metadata key and assigns the value to **@Metadata0**. Subsequently, if record # 12 needs reprinting, the **RunListLink** can be modified to appear as:

TBD 2.x Example.

```

<RunListLink Usage="Input" rRef="MyVDPRunList" ProcessUsage="Document">
  <Part DocTags="CoverLetter" Metadata0="12"/>
  <Part DocTags="Brochure" Metadata0="12"/>
  <Part DocTags="Postcard" Metadata0="12"/>
</RunListLink>

```

8.83.2 Element: ByteMap

A **ByteMap** represents a raster of image data. This data MAY have multiple bits per pixel, MAY represent a varying set of color planes, and MAY be interleaved. A **Bitmap** is a special case of a **ByteMap** in which each pixel is represented by a single bit per color.

Table 8-177: ByteMap Element

Name	Data Type	Description
<i>BandOrdering</i> ?	enumeration	Identifies the precedence given when ordering the produced bands. <i>@BandOrdering</i> is REQUIRED for non-interleaved data and SHALL be ignored for interleaved data if specified. Allowed values are: <i>BandMajor</i> – The position of the bands on the page is prioritized over the color. <i>ColorMajor</i> – All bands of a single color are played in order before progressing to the next plane. This is only possible with non-interleaved data.
<i>FrameHeight</i> ?	integer	Height of the overall image that MAY be broken into multiple bands.
<i>FrameWidth</i> ?	integer	Width of overall image that MAY be broken into multiple columns.
<i>Halftoned</i> ?	boolean	Indicates whether or not the data has been halftoned.
<i>Interleaved</i> ?	boolean	If "true", the data are interleaved or chunky. Otherwise the data are non-interleaved or planar.
<i>PixelColorants</i> ?	NMTOKENS	Ordered list of separation names containing information about which colorants are represented.
<i>PixelDepth</i> ?	integer	Number of bits per pixel for each colorant.
<i>PixelSkip</i> ?	integer	Number of bits to skip between pixels of interleaved data.
<i>Resolution</i> ?	XYPair	Output resolution.
<i>Band</i> ?	element	Description of the structure of the bands containing raster data. Multiple bands MAY be referenced by partitioning the parent RunList .

8.83.3 Element: Band

Table 8-178: Band Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Height</i> ?	integer	Height in pixels of the band.

Table 8-178: Band Element (Sheet 2 of 2)

Name	Data Type	Description
WasMarked?	boolean	Indicates whether any rendering marks were made in this band. This Attribute allows a band to be skipped if no marks were made in the band.
Width?	integer	Width in pixels of the band

— Attribute: ElementType

Table 8-179: ElementType Attribute Values

Value	Description
Auxiliary	Any type of file that is needed to complete a layout but not explicitly displayed (e.g., ICC profiles or fonts).
Barcode	A barcode.
Composed	Combination of elements that define an element that is not bound to a document page.
Document	An ordered set of one or more pages.
Graphic	Line art.
IdentificationField	A general identification field excluding bar codes.
Image	Bitmap image.
MultiDocument	An ordered set of one or more Documents including document breaks (e.g., PPML, PPML/VDX).
MultiSet	An ordered set of one or more document sets, including document set breaks, document breaks and sub document breaks (e.g., PPML, PPML/VDX, ISO 16612-2 PDF/VT)
Page	Representation of one document page.
Reservation	Empty element. Content for this area of the page might be provided by a subsequent Process.
Surface	Representation of an imposed surface.
Text	Formatted or unformatted text.
Tile	Representation of the contents of one tile.

Example 8-30: RunList: Unstructured Single-File RunList

The following five examples illustrate how a **RunList** can be structured using Partitioning mechanisms. Note that the Partitioning of a **RunList** often generates the values necessary to evaluate the Partitioning of other Resources (e.g., the **@RunIndex** into the **RunList**). Thus, the order in which the **RunList** Elements appear in the XML document is significant. Note that the **@Run** Partition Key has a string value, which MAY be non-numeric. Below is an example of simple unstructured single-file **RunList**. This example specifies all pages contained in *"/in/colortest.pdf"*.

TBD 2.x Example.

```
<RunList Class="Parameter" ID="Link0003" Pages="0 ~ -1" Status="Available">
  <LayoutElement>
    <FileSpec URL="File:///in/colortest.pdf"/>
  </LayoutElement>
</RunList>
```

Example 8-31: RunList: Multi-File Unseparated RunList

Example of simple multi-file unseparated **RunList** using **RunList/@Directory**. This example specifies all pages contained in File1.pdf and File2.pdf, which are located in the directory "///Dir/" that is specified in **RunList/@Directory**.

TBD 2.x Example.

```
<RunList Class="Parameter" Directory="File:///Dir/" ID="Link0003"
    PartIDKeys="Run" Status="Available">
    <RunList Pages="0 ~ -1" Run="1">
        <LayoutElement>
            <FileSpec URL="File1.pdf"/>
        </LayoutElement>
    </RunList>
    <RunList Pages="0 ~ -1" Run="2">
        <LayoutElement>
            <FileSpec URL="File2.pdf"/>
        </LayoutElement>
    </RunList>
</RunList>
```

Example 8-32: RunList: Multi-File Unseparated RunList with Spawning

Example of simple multi-file unseparated **RunList** with independent spawning. This example specifies the first five pages contained in File1.pdf and File2.pdf. File2.pdf has been spawned and is being processed individually.

TBD 2.x Example.

```
<RunList Class="Parameter" ID="Link0003" PartIDKeys="Run" Status="Available">
    <RunList Pages="0 ~ 4" Run="1">
        <LayoutElement>
            <FileSpec URL="File:///File1.pdf"/>
        </LayoutElement>
    </RunList>
    <RunList Pages="0 ~ -1" Run="2" SpawnStatus="SpawnedRW">
        <LayoutElement>
            <FileSpec URL="File:///File2.pdf"/>
        </LayoutElement>
    </RunList>
</RunList>
```

Example 8-33: RunList: Spawner RunList

This is the corresponding spawner **RunList**. Note the **@LogicalPage** Attribute, which specifies the number of skipped pages.

TBD 2.x Example.

```
<RunList Class="Parameter" ID="Link0003" LogicalPage="5" Pages="0 ~ -1"
    PartIDKeys="Run" Status="Available">
    <RunList Run="2">
        <LayoutElement>
            <FileSpec URL="File:///File2.pdf"/>
        </LayoutElement>
    </RunList>
</RunList>
```

Example 8-34: RunList: Multi-File Separated RunList

This example specifies all pages contained in Presep.pdf and following that, pages 1, 3 and 5 of each preseparated file.

TBD 2.x Example.

```
<RunList Class="Parameter" ID="Link0003" PartIDKeys="Run Separation"
    Status="Available">
    <RunList Run="1" SkipPage="3">
        <LayoutElement>
            <FileSpec URL="File:///Presep.pdf"/>
        </LayoutElement>
        <RunList FirstPage="0" IsPage="false" Separation="Cyan"/>
        <RunList FirstPage="1" IsPage="false" Separation="Magenta"/>
        <RunList FirstPage="2" IsPage="false" Separation="Yellow"/>
        <RunList FirstPage="3" IsPage="false" Separation="Black"/>
    </RunList>
    <RunList IsPage="true" Pages="1 3 5" Run="2">
        <RunList IsPage="false" Separation="Cyan">
            <LayoutElement>
                <FileSpec URL="File:///Cyan2.pdf"/>
            </LayoutElement>
        </RunList>
        <RunList IsPage="false" Separation="Magenta">
            <LayoutElement>
                <FileSpec URL="File:///Magenta2.pdf"/>
            </LayoutElement>
        </RunList>
        <RunList IsPage="false" Separation="Yellow">
            <LayoutElement>
                <FileSpec URL="File:///Yellow2.pdf"/>
            </LayoutElement>
        </RunList>
        <RunList IsPage="false" Separation="Black">
            <LayoutElement>
                <FileSpec URL="File:///Black2.pdf"/>
            </LayoutElement>
        </RunList>
    </RunList>
</RunList>
```

8.84 ScavengerArea

This Resource describes a scavenger area for removing excess ink from printed Sheets. It is defined within a **MarkObject** of a surface.

Resource Properties

Resource referenced by: Layout/MarkObject

Input of Processes: —

Output of Processes: —

Table 8-180: ScavengerArea Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>Center</i>	XYPair	Position of the center of the scavenger area in the coordinates of the MarkObject that contains this mark.
<i>Rotation ?</i>	double	Rotation in degrees. Positive graduation figures indicate counter-clockwise rotation; negative figures indicate clockwise rotation.
<i>Separations ?</i>	NMTOKENS	Set of separations to which the scavenger area is bound.

Table 8-180: ScavengerArea Resource (Sheet 2 of 2)

Name	Data Type	Description
Size ?	XYPair	Size of the scavenger area.

8.85 ScreeningParams

This Resource specifies the parameter of the **Screening** Process. Since screening is, in most cases, very OEM specific, the following parameters are generic enough that they can be mapped onto a number of OEM controls.

Resource Properties

Resource referenced by: **Content, ExposedMedia, , RunList**

Intent Pairing: **ContentCheckIntent,**

Input of Processes: **ContoneCalibration, Screening**

Output of Processes: —

Table 8-181: ScreeningParams Resource

Name	Data Type	Description
<i>IgnoreSourceFile</i> ?	boolean	Specifies whether to ignore the screen settings (e.g., setscreen, setcolorscreen and sethalftone) specified in the source files. Note that in some cases, halftones are used to create patterns. In these cases, the halftone in the source PDL file will not be overridden.
ScreenSelector *	element	List of screen selectors. A screen selector is included for each separation, including a default specification. ScreenSelector SHALL contain the complete set of Parameters for a given screening operation. For instance, it is invalid to specify one ScreenSelector for a given <i>@ObjectTags</i> and another ScreenSelector for a given <i>@SourceObjects</i> .

8.86 SeparationControlParams

This Resource provides the controls needed to separate composite color files.

Resource Properties

Resource referenced by: —

Intent Pairing: **ContentCheckIntent,**

Input of Processes: **Separation**

Output of Processes: —

Table 8-182: SeparationControlParams Resource

Name	Data Type	Description
AutomatedOverPrintParams ?	element	Controls for overprint substitutions. The default case is that no automated overprint generation is used.

8.87 Shape

Resource Properties

Resource referenced by: **ShapeCuttingParams, ShapeDef**

Input of Processes: —
Output of Processes: —

Table 8-183: Shape Resource

Name	Data Type	Description
<i>CutBox</i> ?	rectangle	Specification of a rectangular window.
<i>CutOut</i> ?	boolean	If " <i>true</i> ", the inside of a specified shape will be removed. If " <i>false</i> ", the outside of a specified shape will be removed. An example of an inside shape is a window. An example of an outside shape is a shaped greeting card.
<i>CutPath</i> ?	PDFPath	Specification of a complex path. This MAY be an open path in the case of a single line.
<i>DDESCutType</i> ?	integer	Type of cut or perforation used. Values include: a number between "0" and "999" corresponding to a line type as defined in DDES3 standard (ANSI® IT8.6-2002). Note: A value of 101 corresponds to a cut line.
<i>ShapeDepth</i> ?	double	Depth of the shape cut, measured in microns [μm]. If not specified, the shape is completely cut.
<i>ShapeType</i>	enumeration	Describes any precision cutting other than hole making. Allowed values are: <i>Path</i> <i>Rectangular</i> <i>Round</i> <i>RoundedRectangle</i> – Rectangle with rounded corners.
<i>TeethPerDimension</i> ?	double	Number of teeth in a given perforation extent, in teeth/point. MicroPerforation is defined by specifying a large number of teeth ($n > 1000$).

8.88 ShapeCuttingParams

ShapeCuttingParams defines the details of the **ShapeCutting** Process.

Resource Properties

Resource referenced by: —

Intent Pairing: **ShapeCuttingIntent**

Input of Processes: **ShapeCutting**

Output of Processes: —

Table 8-184: ShapeCuttingParams Resource

Name	Data Type	Description
<i>DeliveryMode</i> ?	enumeration	<p>Allowed values are:</p> <p><i>FullSheet</i> – The output of the die-cutter are complete Sheets. The blanks are kept in place with nicks. Front waste (gripper margin) has not been removed.</p> <p><i>RemoveGripperMargin</i> – The output of the die-cutter are complete Sheets. The blanks are kept in place with nicks. Front waste (gripper margin) has been removed.</p> <p><i>SeparateBlanks</i> – The output of the die-cutter are blanks that have been removed from the Sheets.</p>
<i>DieLayoutRef</i> ?	IDREF	A Resource containing the reference of an external file describing the cutting and other paths.
<i>ModuleID</i> ?	NMTOKEN	Identifier of the shape-cutting Module in the Press. See ConventionalPrintingParams .
<i>SheetLay</i> ?	enumeration	<p>Lay of input media. Reference edge of where paper is placed in the feeder.</p> <p>Allowed values are:</p> <p><i>Center</i></p> <p><i>Left</i></p> <p><i>Right</i></p>
<i>Shape</i> *	element	Details of each individual cut shape

8.89 ShapeDef

A structural design describing a 2D surface with paths that describe different finishing operations like cutting, creasing, perforation, etc. In the case of box production this Resource is a description of the unprinted blank box as it will be available after die cutting and blanking and before folding. A **ShapeDef** is defined either by an external file (**FileSpec**) describing the structural design or a collection of PDFPaths contained in Shape elements. In case this description is stored in a file, the format of this file may be a vendor specific format, a standard DDES3 file (ANSI® IT8.6-2002), or less well specified but commonly used formats like CFF2 or DXF or even a PDF or EPS file.

Resource Properties

Resource referenced by: *DieLayout/Station, LayoutElementProductionParams*

Input of Processes: *DieLayoutProduction*

Output of Processes: *ShapeDefProduction*

Table 8-185: ShapeDef Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>Area</i> ?	double	The net area of the shape after cutting. (m ²)
<i>CutBox</i> ?	rectangle	A rectangle describing the bounding box of all cut lines. This is sometimes referred to as the knife to knife dimensions of the BlankBox. This Attribute is usually only valid after the generation of the structural design.
<i>Dimensions</i> ?	shape	Width <i>x</i> , height <i>y</i> and depth <i>z</i> coordinates of the open 3D shape. For a box, these are the outer dimensions of the opened and potentially filled box (e.g., for palletizing of the final products). Note: compare with <i>@FlatDimensions</i> .

Table 8-185: ShapeDef Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>FlatDimensions</i> ?	shape	Width <i>x</i> , height <i>y</i> and depth <i>z</i> coordinates of the flat 3D shape. For a box, these are the outer dimensions of the glued flat box (e.g., for palletizing of the boxes prior to filling). This corresponds to the Dimensions of the output component of the BoxFolding process. Note: compare with @Dimensions.
<i>FluteDirection</i> ?	enumeration	Intended direction of the flute for this design in the coordinate system defined by @CutBox. This information needs to be taken into account by the DieLayoutProduction Process to give the ShapeDef the correct orientation on the Media . Allowed values are: <i>XDirection</i> – Along the X-axis of the @CutBox coordinate system. <i>YDirection</i> – Along the Y-axis of the @CutBox coordinate system.
<i>GrainDirection</i> ?	enumeration	Intended direction of the grain for this design in the coordinate system defined by @CutBox. This information needs to be taken into account by the DieLayoutProduction Process to give the ShapeDef the correct orientation on the Media . Allowed values are: <i>XDirection</i> – Along the X-axis of the @CutBox coordinate system. <i>YDirection</i> – Along the Y-axis of the @CutBox coordinate system. <i>Both</i> – Both orientations are acceptable.
<i>MediaRef</i> ?	IDREF	Media for which this structural design was intended for. The Media description defines important design parameters as the type of Media , thickness, inside loss, outside gain, etc. Media /@ <i>GrainDirection</i> and Media /@ <i>FluteDirection</i> do not have any significance.
<i>MediaSide</i> ?	enumeration	Determines the printing side for which the structural design is made. Allowed values are: <i>Front</i> – for a box this corresponds to the outside of a box. <i>Back</i> - for a box this corresponds to the inside of a box. Note: folding carton is usually cut from the outside (Front), corrugated from the inside (Back).
<i>ResourceWeight</i> ?	double	The weight of the shape after cutting (g).
<i>CutLines</i> ?	NMTOKENS	Selects the die line separations from the file referenced by FileSpec . Additional details of the usage of the Separations MAY be specified in the respective ResourceSet [@Name="Color"].
FileSpec ?	element	The FileSpec of the structural design file. The format of this file may be a vendor specific format, a standard DDES3 file (ANSI® IT8.6-2002), less well specified but commonly used formats like CFF2 or DXF or even a PDF or EPS file. FileSpec and Shape are mutually exclusive
Shape *	element	The shape is defined by a collection of Shape Elements. Shape and FileSpec are mutually exclusive.

8.90 ShapeDefProductionParams

Parameters for the structural design.

Resource Properties

Resource referenced by: —

Input of Processes: *ShapeDefProduction*

Output of Processes: —

Table 8-186: ShapeDefProductionParams Resource

Name	Data Type	Description
ObjectModel *	element	A 3D model of the objects that need to be packed.
ShapeTemplate ?	element	A structural template sometimes called a parametric structural design. Given a set of parametric values a structural template can be instantiated to an actual structural design.

8.90.1 Element: ObjectModel

Table 8-187: ObjectModel Element

Name	Data Type	Description
Dimensions ?	shape	Width <i>x</i> , height <i>y</i> and depth <i>z</i> values for the bounding box of the object.
FileSpec ?	element	The FileSpec of the 3D model of the objects that needs to be packed. The format of this file may be a vendor specific format or a standard 3D format like VRML or PDF (U3D).

8.90.2 Element: ShapeTemplate

ShapeTemplate describes a structural template which is also referred to as a parametric structural design.

Table 8-188: ShapeTemplate Element

Name	Data Type	Description
InnerDimensions ?	shape	Width <i>x</i> , height <i>y</i> and depth <i>z</i> coordinates of the 3D shape. For a box these are the inner dimensions.
Name ?	string	The name of a parametric structural design or CAD template.
Standard ?	NMTOKEN	The name of the standard this template belongs to (e.g., FEFCO, ECMA or the name of a company internal standard).
FileSpec ?	element	The FileSpec of the parametric structural design.
ShapeDimension *	element	ShapeDimension elements define additional parametric values of the ShapeTemplate.

The three Figures below show shapes specified by a ShapeTemplate with each named variable represented by a ShapeDimension element Example 8-49 below illustrates the encoding of L,D and W using ShapeDimension:

8.90.3 Element: ShapeDimension

Table 8-189: ShapeDimension Element

Name	Data Type	Description
<i>Usage</i>	string	@Usage specifies the name of the ShapeDimension. Values Include: <i>D</i> - Depth <i>L</i> - Length <i>W</i> - Width
<i>Value</i>	double	@Value specifies the length of the ShapeDimension in points.

Example 8-35: ShapeTemplate for Figure 8-55

TBD 2.x Example.

```
<ShapeDefProductionParams Class="Parameter" ID="Link0003"
    Status="Available">
    <ShapeTemplate>
        <GeneralID IDUsage="L" DataType="double" IDValue="1440.0"/>
        <GeneralID IDUsage="W" DataType="double" IDValue="720.0"/>
        <GeneralID IDUsage="D" DataType="double" IDValue="1440.0"/>
    </ShapeTemplate>
</ShapeDefProductionParams>
```

Figure 8-44: ShapeTemplate Example 1

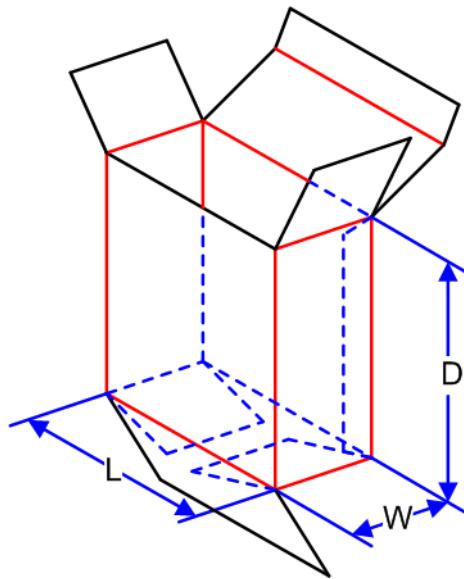


Figure 8-45: ShapeTemplate Example 2

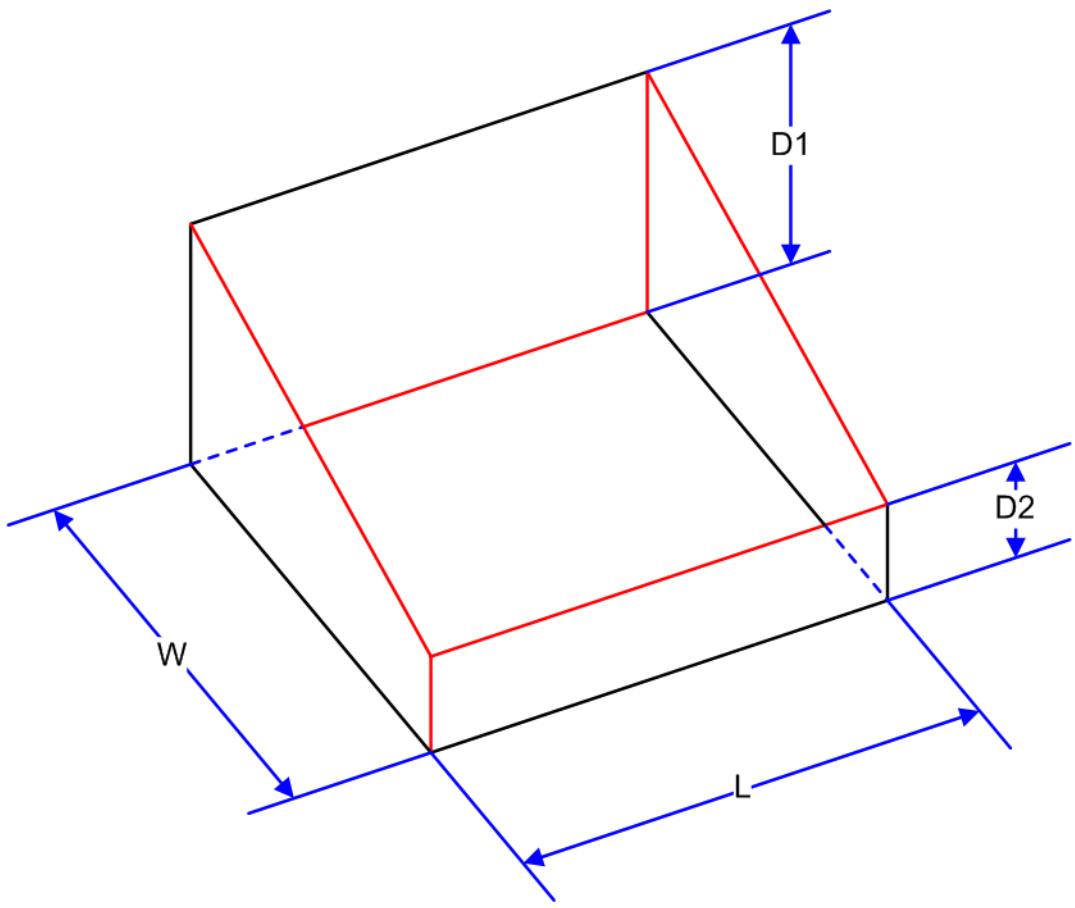
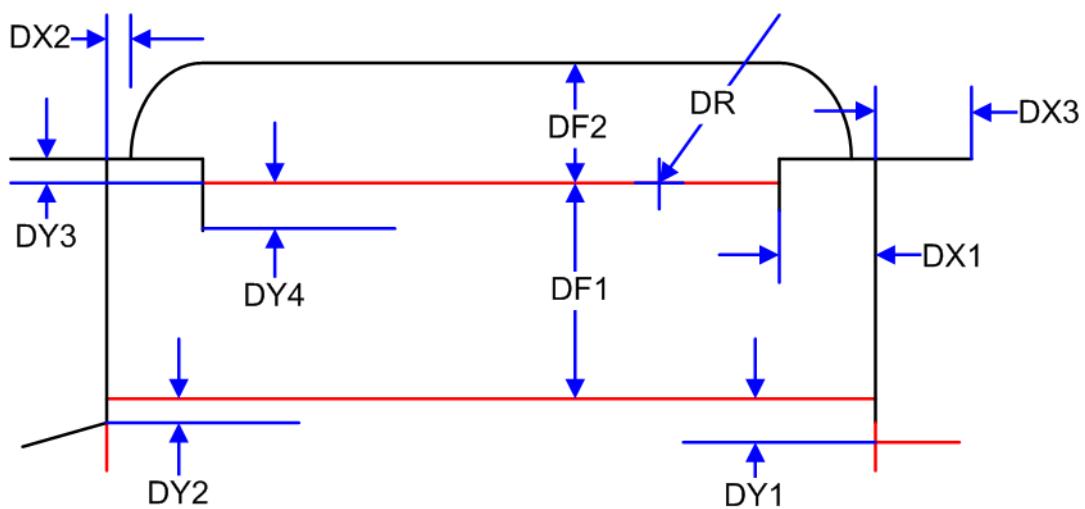


Figure 8-46: ShapeTemplate Example 3



8.91 SheetOptimizingParams

Parameter resource that parametrizes the ***SheetOptimizing*** process.

Resource Properties

Resource referenced by: —

Input of Processes: ***SheetOptimizing***

Output of Processes: —

Table 8-190: SheetOptimizingParams Resource

Name	Data Type	Description
ConvertingConfig +	element	Specification of the device configurations for destination sheet sizes.
GangElement +	element	Each GangElement describes an individual product or product part that SHALL be placed completely on a gang form. If an individual Product MAY be distributed over multiple separate gangs (e.g., cover and body with different paper), it SHALL be represented as multiple gang elements.

8.91.1 Element: GangElement

A GangElement describes an individual product or product part (e.g., product cover) that is a candidate for placement on a printed sheet.

Table 8-191: GangElement Element (Sheet 1 of 3)

Name	Data Type	Description
<i>BinderySignatureIDs</i> ?	NMTOKENS	The @AssemblyIDs of the partitions that may be ganged. Any StrippingParams partitions in the GangElement that have an Assembly ID that is not in this list SHALL NOT be ganged. If not specified all partitions are selected. If specified, @BinderySignatureIDs SHALL list the BinderySignature/@BinderySignatureID of all BinderySignatures that are candidates for this GangElement.
<i>CollapseBleeds</i> ?	boolean	If single page GangElement that has a bleed in a solid color is ganged in block pattern, the bleed between the GangElement elements may not be required. If "true", the bleed margin between the instances of GangElement elements SHOULD be removed.
<i>Dimension</i> ?	XYPair	The GangElement block size including trims and bleeds of the element to be ganged. SHALL NOT be specified if @NPage is specified. If GangElement/StrippingParams is specified @Dimension SHALL NOT be specified.
<i>DueDate</i> ?	dateTime	The latest date and time the GangElement needs to be included on a gang. The gang engine SHOULD use a combination of @DueDate and @Priority to decide which GangElement Elements to place on a gang.
<i>ExternalID</i> ?	NMTOKEN	The product ID in (e.g., a web to print system).
<i>FillPriority</i> ?	integer	If non-zero the ganging engine is requested to fill any left over space on the sheet with this GangElement Elements even if this would lead to over production of the GangElement Elements. GangElement Elements with a higher priority take precedence over GangElement Elements with a lower priority.

Table 8-191: GangElement Element (Sheet 2 of 3)

Name	Data Type	Description
<i>GangElementID</i>	ID	The ID of the GangElement that is unique within the context of the Workflow. The <i>@GangElementID</i> SHALL be copied from the Input GangElement/StrippingParams partitions to the Output StrippingParams partitions to indicate which GangElement have been included in the results of the <i>SheetOptimizing</i> process.
<i>GrainDirection ?</i>	enumeration	The allowed grain direction of the paper with respect to the GangElement. <i>@GrainDirection</i> is specified in the context of the page. If no page context exists, then <i>@GrainDirection</i> references the entire rectangle. Allowed values are from: MediaIntent/ <i>@GrainDirection</i> . See Table 7.15, “MediaIntent” on page 500.
<i>GroupCode ?</i>	NMTOKEN	Code specifying a group of products. GangElement elements with the same group code MAY be ganged together in a vertical column on the sheet, whereas GangElement elements with different <i>@GroupCode</i> values SHOULD NOT be grouped. This attribute MAY be used to prevent GangElement elements with different colors, ink densities or other incompatible properties to be placed in vertical columns when doing offset printing.
<i>JobID ?</i>	NMTOKEN	The original <i>@JobID</i> of the element to be ganged.
<i>LayoutRef?</i>	IDREF	Layout that describes the list of FOLDING sheets outside of a sheet context for this GangElement. Only partition by BinderySignature .
<i>MaxQuantity ?</i>	integer	The maximum number of printed (fold) sheets that may be produced by the gang (including finishing waste).
<i>MediaRef?</i>	IDREF	The characteristics of the target Media that SHALL be met in the gang.
<i>MinQuantity ?</i>	integer	The minimum number of printed (fold) sheets that SHALL be produced by the gang (including finishing waste).
<i>NPage ?</i>	integer	The total number of pages of the GangElement. If GangElement/StrippingParams is specified, then <i>@NPage</i> SHALL NOT be specified. If <i>@NPage</i> is specified, the number and size of the fold sheets / BinderySignature elements is decided by the ganging engine.
<i>NumberUp ?</i>	XYPair	The number up that is to be placed on the gang in a single block. If Y is zero, that other number sets the total number-up requested without specifying a specific number in X or Y direction.
<i>NumColors ?</i>	XYPair	The first value specifies the number of colors on the front side. The second value specifies the number of colors on the back side. The value 0 implies no print on the respective side. The value 1 implies Black. The value 4 implies CMYK. If both SeparationListFront or SeparationListBack and <i>@NumColors</i> are specified. The implied values from <i>@NumColors</i> SHALL be added to the respective SeparationListFront or SeparationListBack when evaluating.

Table 8-191: GangElement Element (Sheet 3 of 3)

Name	Data Type	Description
<i>OneSheet</i> ?	NMTOKEN	Control how this GangElement SHOULD be placed on ganged sheets. Values include: <i>Any</i> – Place on any sheet that is generated. <i>GangElementID</i> – Keep all blocks with this <i>@GangElementID</i> on one sheet. <i>JobID</i> – Keep all GangElement Elements with the same JobID on the same sheet.
<i>OrderQuantity</i>	integer	The number of printed (fold) sheets to produce (including finishing waste).
<i>PageDimension</i> ?	XYPair	The GangElement page size or page size (including trims and bleeds?) of the element to be ganged. MAY be specified if <i>@NPage</i> is specified. If GangElement/StrippingParams is specified <i>@PageDimension</i> SHALL NOT be specified.
<i>Priority</i> ?	integer	All GangElement elements with a <i>@Priority</i> = "100" SHALL be included in the gang. GangElement elements with a <i>@Priority</i> less than 100 MAY be included in the gang, and SHOULD be in descending <i>@Priority</i> order.
<i>RotationPolicy</i> ?	enumeration	Specifies the level of freedom when applying the values specified in <i>@GrainDirection</i> Allowed values are: <i>Exact</i> – A 180 degree rotation is not allowed <i>Free</i> – A 180 degree rotation is allowed.
<i>RunListRef</i> ?	IDREF	Reference to the content data for this GangElement. If this RunList refers to a structured PDL with multiple document instances such as recipient records in PDF/VT, then this GangElement represents multiple individual sections. These sections SHALL be positioned using the same rules that would apply if each document instance were referenced by an individual GangElement. All Document instances referenced by an individual GangElement SHALL be processed in one gang.

8.91.2 Element: SeparationListBack

Separation List for Back.

Table 8-192: SeparationListBack Element

Name	Data Type	Description
<i>Separations</i>	NMTOKENS	Description of the separations used.

8.91.3 Element: SeparationListFront

8.92 Separation List for Front.**ShrinkingParams**

Table 8-193: SeparationListFront Element

Name	Data Type	Description
<i>Separations</i>	NMTOKENS	Description of the separations used.

This Resource provides the parameters for the **Shrinking** Process in shrink wrapping.

Resource Properties

Resource referenced by: —

Intent Pairing:

Input of Processes: **Shrinking**

Output of Processes: —

Table 8-194: ShrinkingParams Resource

Name	Data Type	Description
<i>Duration</i> ?	duration	Shrinking time.
<i>ShrinkingMethod</i> ?	enumeration	Specifies of the shrinking method for shrink wrapping. Allowed values are: <i>ShrinkCool</i> <i>ShrinkHot</i>
<i>Temperature</i> ?	double	Oven temperature in ° Centigrade.

8.93 SpinePreparationParams

SpinePreparationParams describes the preparation of the spine of book blocks for hard and soft cover book production (e.g., milling and notching).

Resource Properties

Resource referenced by: —

Intent Pairing: **BindingIntent**

Input of Processes: **SpinePreparation**

Output of Processes: —

Table 8-195: SpinePreparationParams Resource (Sheet 1 of 2)

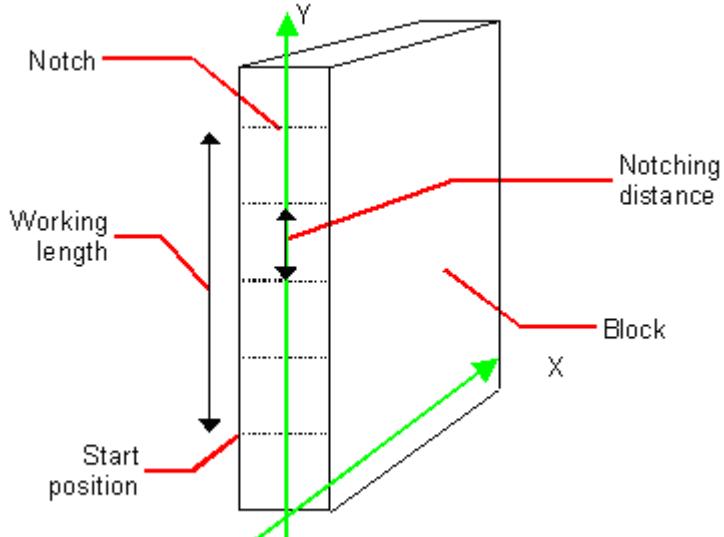
Name	Data Type	Description
<i>MillingDepth</i> ?	double	Milling depth, in points. This describes the total cut-off of the spine, regardless of the technology used to achieve this goal.
<i>NotchingDepth</i> ?	double	Notching depth relative to the leveled spine, in points. If not specified, there is no notching.
<i>NotchingDistance</i> ?	double	Notching distance, in points.
<i>Operations</i> ?	NMTOKENS	List of operations to be applied to the spine. Duplicate entries are allowed to specify a sequence of identical operations. The order of operations is significant. Values include those from: Table 8-291, “Operations Attribute Values”.

Table 8-195: SpinePreparationParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>StartPosition</i> ?	double	Starting position of milling tool along the Y-axis of the operation coordinate system.
<i>WorkingLength</i> ?	double	Working length of milling operation. If specified larger than the spine length, the complete spine is prepared. If not specified, the complete spine is prepared.

— Attribute: Operations**Table 8-196: Operations Attribute Values**

Value	Description
<i>Brushing</i>	Brushes away dust from the spine to improve the binding quality.
<i>FiberRoughing</i>	The fibers of the paper on the spine are exposed without the risk of glazing the paper coating. This optimizes the spine preparation considering paper and adhesive types.
<i>Leveling</i>	After milling the spine, any uneven areas are leveled to achieve an even surface.
<i>Milling</i>	Cuts off part of the spine so the spine is not too even. A rough texture of the fibers is assured. This creates ideal conditions for stable anchoring of the Sheets in the glue.
<i>Notching</i>	This gives a clamping effect on the spine which is desirable for some products.
<i>Sanding</i>	Is used for voluminous book papers.
<i>Shredding</i>	Produces a relatively smooth surface. Further operations like " <i>Notching</i> ", " <i>Leveling</i> ", " <i>FiberRoughing</i> ", " <i>Sanding</i> " or " <i>Brushing</i> " are necessary.

Figure 8-47: Parameters and coordinate systems for the SpinePreparation Process

8.94 SpineTapingParams

SpineTapingParams define the parameters for taping a strip tape or kraft paper to the spine of a book block.

Resource Properties

Resource referenced by:

Intent Pairing:

BindingIntent

Input of Processes:

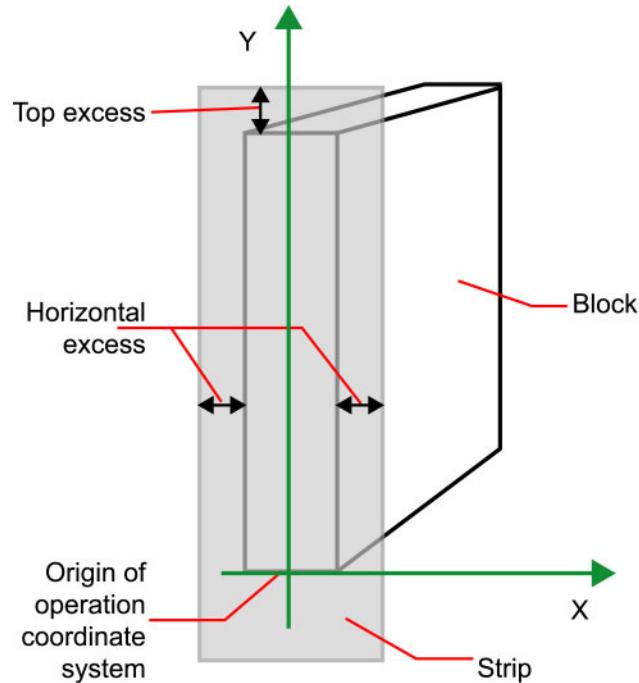
SpineTaping

Output of Processes:

—

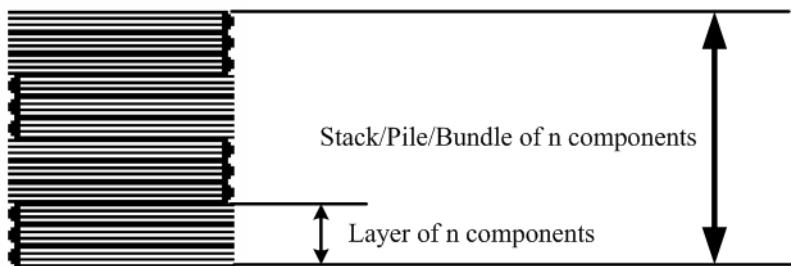
Table 8-197: SpineTapingParams Resource

Name	Data Type	Description
<i>HorizontalExcess</i> ?	double	Taping spine excess on each side. The tape is assumed to be centered between left and right.
<i>HorizontalExcessBack</i> ?	double	Horizontal excess of back if tape is not centered
<i>StripBrand</i> ?	string	Strip brand.
<i>StripColor</i> ?	NamedColor	Color of the strip.
<i>StripColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If <i>@StripColorDetails</i> is supplied, <i>@StripColor</i> SHOULD also be supplied.
<i>StripLength</i> ?	double	Length of strip material along binding edge. If not defined, the default case is that the <i>@StripLength</i> be equivalent to the length of the spine.
<i>StripMaterial</i> ?	enumeration	Strip material. Allowed values are: <i>Calico</i> <i>Cardboard</i> <i>CrepePaper</i> <i>Gauze</i> <i>Paper</i> <i>PaperlinedMules</i> <i>Tape</i>
<i>TopExcess</i> ?	double	Top spine taping excess. This value MAY be negative.
<i>GlueLine</i> *	element	Describes where and how to apply glue to the book block.

Figure 8-48: Parameters and coordinate system for the SpineTaping Process

8.95 StackingParams

Settings for the **Stacking** Process. A stack of components might be uneven and unstable, due to variations in thickness across each component. The thickness variations might be caused by folding, binding or inserted components. A stack might be split into layers, with successive layers rotated by 180° to compensate for the unevenness (Figure 8-60).

Figure 8-49: Stacking Layers

If the thickest part is on an edge (e.g., a book binding), the components might be offset to separate the thick parts. Layer compensation and offsetting can be combined as in the following examples of pile patterns (Figure 8-61).

Figure 8-50: Pile Patterns

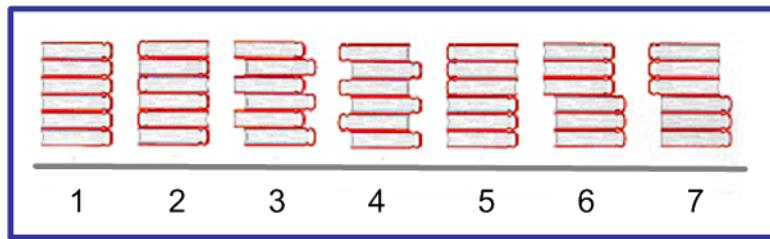


Table 8-198: Parameters in Stacking

Pile Pattern	StandardAmount	LayerAmount (Default = StandardAmount)	Compensate (Default = true)	Disjoining/ @Offset
1	"6"	"6"	"true"	"0 0"
2	"6"	"1"	"true"	"0 0"
3	"6"	"1"	"false"	"x 0"
4	"6"	"1"	"true"	"x 0"
5	"6"	"3"	"true"	"0 0"
6	"6"	"3"	"false"	"x 0"
7	"6"	"3"	"true"	"x 0"

If the number of components is not evenly divisible by standard stack size (`@StandardAmount`) or the number of components in a bundle is not evenly divisible by layer size (`@LayerAmount`), there will be a remainder, yielding one or more odd-count stacks or layers. By default, the odd-count stack or layer size can contain as few as one component. This might exceed equipment cycle times, and flimsy components (newspapers) might cause problems with downstream equipment such as strappers. The `@MinAmount` and `@MaxAmount` control the minimum and maximum size of odd-count stacks and layers. The following figures show the odd count handling for bundles and layers.

Figure 8-51: Odd count handling for a Bundle

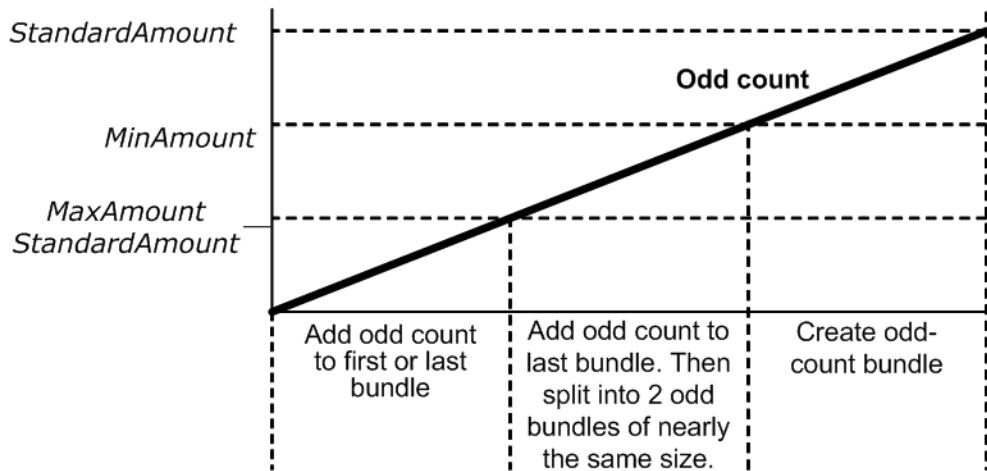
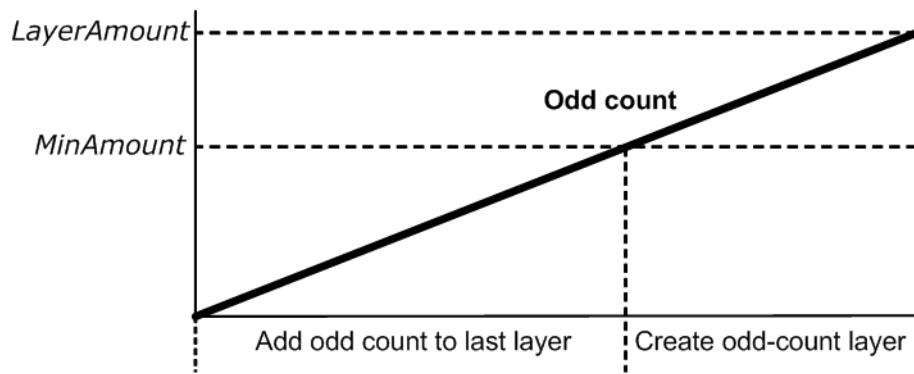


Figure 8-52: Odd count handling for a Layer**Resource Properties**

Resource referenced by:

Intent Pairing:

Input of Processes: *Stacking*

Output of Processes: —

Table 8-199: StackingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>BundleDepth</i> ?	integer	In case of nested bundles with <i>@BundleType = "Stack"</i> , this parameter addresses the Element to be consumed within the “tree” of such bundles to allow a level of de-stacking. If the real bundle depth level (<i>@BundleType = "Stack"</i>) is smaller than the value of <i>@BundleDepth</i> , individual stack items (i.e., the smallest available level) shall be consumed. If the Input Component referenced does not contain bundles, then this parameter is ignored. <i>@BundleDepth = "0"</i> addresses the entire Component , <i>@BundleDepth = "1"</i> addresses the bundle in the Component and so on.
<i>Compensate</i> ?	boolean	180 degree rotation applied to successive layers to compensate for uneven stacking. If <i>@LayerAmount = @StandardAmount</i> , there is one layer, and effectively no compensation.
<i>LayerAmount</i> ?	IntegerList	Ordered number of products in a layer. The first number is the first <i>@LayerAmount</i> , etc. If there are more layers than entries in the list, counting restarts at the first entry. The sum of all entries is typically an even divisor of <i>@StandardAmount</i> . When not known, the default case is that the value of <i>@LayerAmount</i> be equivalent to the value of <i>@StandardAmount</i> .
<i>LayerCompression</i> ?	boolean	If true layer is compressed before next layer is started.
<i>LayerLift</i> ?	boolean	If true layer is lifted to reduce height.
<i>MaxAmount</i> ?	integer	Maximum number of products in a stack, <i>@MaxAmount >= @StandardAmount</i> . When not known, the default case is that the value of <i>@MaxAmount</i> be equivalent to the value of <i>@StandardAmount</i> .
<i>MaxHeight</i> ?	integer	Max height of the stack
<i>MaxWeight</i> ?	double	Maximum weight of a stack in grams.

Table 8-199: StackingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>MinAmount</i> ?	integer	Minimum number of products in a stack or layer, (@MaxAmount – @StandardAmount) <= @MinAmount < @StandardAmount and @MinAmount < @LayerAmount. Where not known, the default case is to use a value equivalent to @MaxAmount – @StandardAmount.
<i>PreStackAmount</i> ?	integer	Amount that is gathered at first
<i>PreStackMethod</i> ?	enumeration	Allowed values are: <i>All</i> – all layers are pre-stacked <i>First</i> – only first layer is pre-stacked <i>None</i> – no pre-stacking
<i>StackCompression</i> ?	boolean	If true stack is compressed before push out
<i>StandardAmount</i> ?	integer	Number of products in a standard stack.
<i>UnderLays</i> ?	IntegerList	Number of underlay Sheets at each layer. The first value is underneath the bottom layer, the next value above the bottom layer and so forth. If more layers than values are specified, counting restarts at the 0 position of @UnderLays. If less layers than values are specified, all underlay Sheets that are not adjacent to a layer are ignored.
<i>Disjointing</i> ?	element	Details of the offset or shift applied to successive layers to separate the thicker portions of components, for example, offsetting the spines of hard-cover books.

8.96 StaticBlockingParams

StaticBlockingParams defines the details of **StaticBlocking**.

Resource Properties

Resource referenced by:

Input of Processes: **StaticBlocking**

Output of Processes: —

Table 8-200: StaticBlockingParams Resource

Name	Data Type	Description
		No attributes defined

8.97 StitchingParams

This Resource provides the parameters for the **Stitching** Process. The process coordinate system is defined as follows:

- The Y-axis increases from the (first) registered edge to the edge opposite to the registered edge.
- The X-axis is aligned with the (second) registered edge, and it increases from the binding edge (or first registered edge) to the edge opposite to the binding edge (or first registered edge).

Note that the stitches are applied from the front in the figures describing the stitching coordinate system.

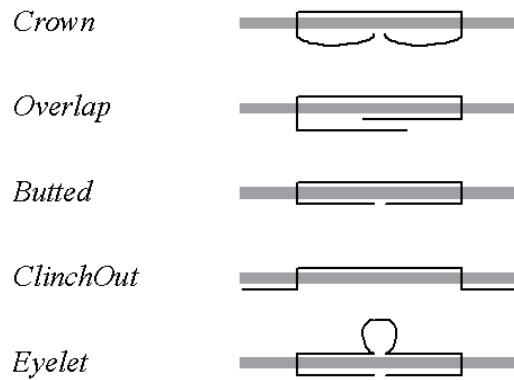
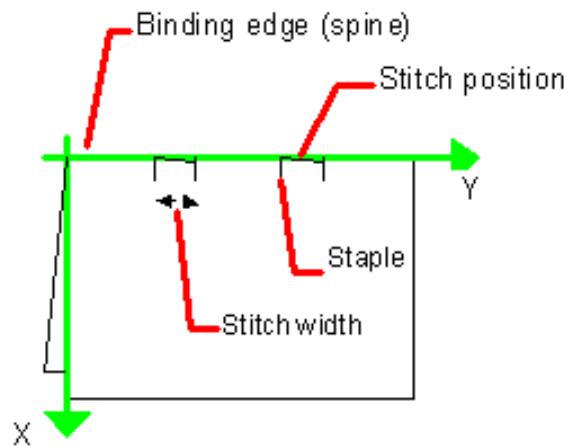
Figure 8-53: Staple shapes**Figure 8-54: Parameters and coordinate system used for saddle stitching**

Figure 8-55: Parameters and coordinate system used for Stitching

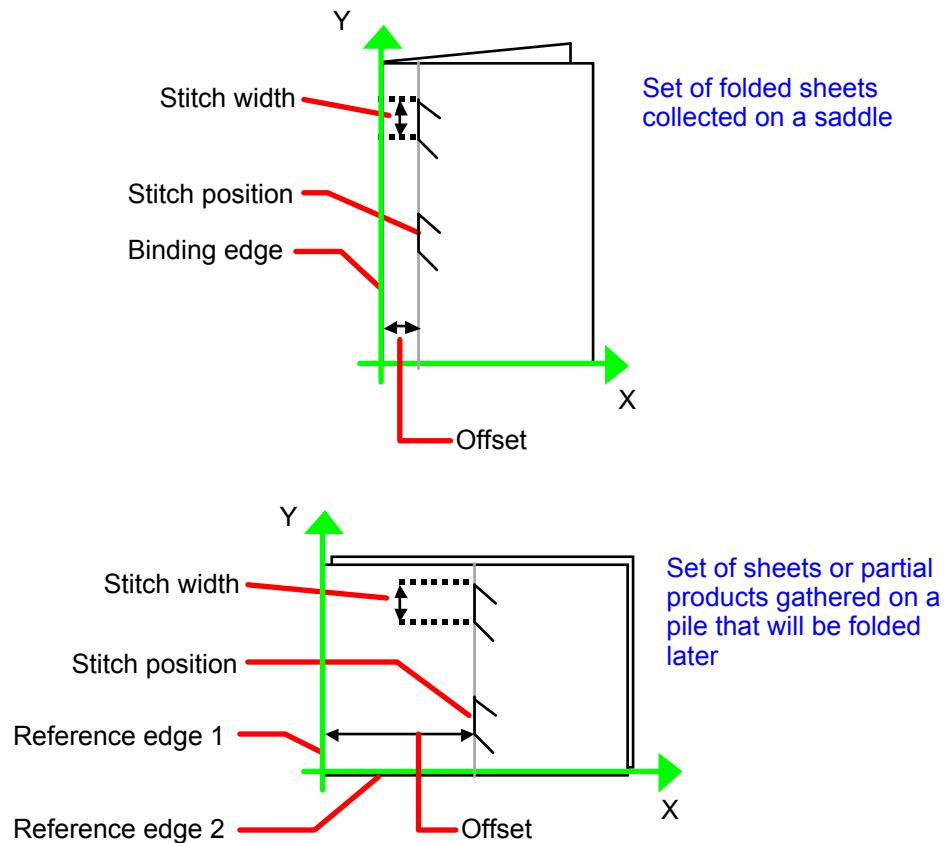
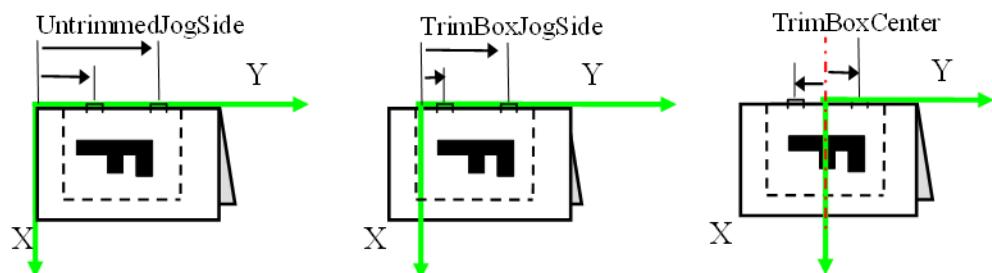


Figure 8-56: Stitching Coordinate System for StitchOrigin Values



Resource Properties

Resource referenced by:

Intent Pairing:

Input of Processes:

Output of Processes:

—

BindingIntent

Stitching

—

Table 8-201: StitchingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>Angle</i> ?	double	Angle of stitch in degree. The angle increases in a counterclockwise direction. Horizontal = "0", which means that it is parallel to the X-axis of the operation coordinate system. Defaults to the system-specified value which MAY vary depending on other Attributes set in this Resource. If @ <i>StitchType</i> = "Saddle", @ <i>Angle</i> SHALL be ignored
<i>NumberOfStitches</i> ?	integer	Number of stitches. If not specified, use the system-specified number of stitches which MAY vary depending on other Attributes set in this Resource. Use a "0" value to use the stitcher without inserting any stitches. Use "NoOp" to bypass the stitcher altogether.
<i>Offset</i> ?	double	Distance between stitch and binding edge. If @ <i>StitchType</i> = "Saddle", @ <i>Offset</i> SHALL be ignored. Note that it is possible to describe saddle stitching with an offset by defining @ <i>StitchType</i> = "Side" with a large @ <i>Offset</i> value.
<i>StapleShape</i> ?	enumeration	<p>Specifies the shape of the staples to be used.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Crown</i> <i>Overlap</i> <i>Butted</i> <i>ClinchOut</i> <i>Eyelet</i> <p>Note: representations of the values are displayed in Figure 8-64.</p>
<i>StitchOrigin</i> ?	enumeration	<p>Defines the origin of @<i>StitchPositions</i>. For an illustration of the values, see Figure 8-67.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>TrimBoxCenter</i> <i>TrimBoxJogSide</i> <i>UntrimmedJogSide</i>
<i>StitchPositions</i> ?	DoubleList	Array containing the stitch positions. The center of the stitch SHALL be specified, and the number of entries SHALL match the number given in @ <i>NumberOfStitches</i> .
<i>StitchType</i> ?	enumeration	<p>Specifies the type of the Stitching operation.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Corner</i> – Stitch in the corner that is at the clockwise end of the reference edge. For example, to stitch in the upper right corner set ComponentLink/@Orientation = "Rotate90". <i>Saddle</i> – Stitch on the middle fold which is on the saddle. <i>Side</i> – Stitch along the reference edge.
<i>StitchWidth</i> ?	double	Width of the stitch to be used. If not present or "0", means use the system-specified width of stitches which MAY vary depending on other Attributes set in this Resource.

Table 8-201: StitchingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>TightBacking</i> ?	enumeration	<p>Definition of the geometry of the back of the product.</p> <p>See BlockPreparationParams/@TightBacking in Section 8.10, “BlockPreparationParams” on page 543 for details.</p> <p>Allowed values in order of increasing pressure are:</p> <ul style="list-style-type: none"> <i>Round</i> – no tight backing is applied <i>RoundBacked</i> – rounding way, backing way <i>Flat</i> <i>FlatBacked</i> – backing way
<i>WireBrand</i> ?	string	Brand of the wire to be used.
<i>WireGauge</i> ?	double	Gauge of the wire to be used. If not present or “0”, means use the system-specified wire gauge which MAY vary depending on other Attributes set in this Resource.
FileSpec (CIP3) ?	element	Reference to a CIP3 file that contains stitching instructions in the CIP3 format.

8.98 StrappingParams

StrappingParams defines the details of **Strapping**.

Resource Properties

Resource referenced by:

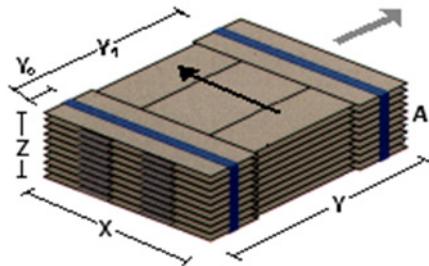
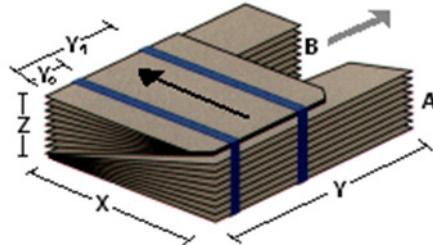
Intent Pairing:

Input of Processes: **Strapping**

Output of Processes: —

Table 8-202: StrappingParams Resource

Name	Data Type	Description
<i>StrappingType</i>	enumeration	<p>Strapping pattern.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Single</i> – One strap. <i>Double</i> – Two parallel single straps. <i>Cross</i> – Two crossed straps. <i>DoubleCross</i> – Two cross straps that strap each side of a box.
<i>StrapPositions</i> ?	NumberList	Positions of the Straps beginning from the origin of the coordinate system (bottom side) increasing from minimum to maximum in points. Each Strap is defined by a 3-tuple of which two values SHALL be 0. The non-zero value specifies the variable coordinate. For instance, two parallel straps shifted along the y-axis are specified as “0 <i>y1</i> 0 0 <i>y2</i> 0” (see Figure 8-68 and Figure 8-69). A centered cross strap in the x-y plane would be specified as “x/2 0 0 0 <i>y/2</i> 0”, which specifies one strap in the x-plane and another in the y-plane.

Figure 8-57: Strapped Bundle**Figure 8-58: Strapped Bundle with Sub-bundles**

8.99 StrippingParams

The **StrippingParams** Resource is a high-level description of how a **Component** is to be produced. It is typically produced by the MIS production planning module and consumed by a prepress workflow system, although its usage is not restricted to this example. There are enough OPTIONAL Attributes to use the same Resource for the interface between estimation systems and production planning systems.

StrippingParams specifies how the surfaces of the **BinderySignature** Elements of a Job are placed onto press Sheets and also gives concrete values for the various **SignatureCell** defined by the **BinderySignature**.

The Partitioning of **StrippingParams** defines the structure of the finished product and the structure of the **Layout** Resource that is produced by the **Stripping** Process. It is therefore RECOMMENDED to Partition the **StrippingParams** Resource by **@SheetName**. Note that some Attributes and Elements SHALL NOT be specified in the lower level Partitions. For instance, **@Device** and **@WorkStyle** are only useful up to the **@SheetName** Partition level.

Resource Properties

Resource referenced by:

—

Intent Pairing:

LayoutIntent, **ContentCheckIntent**

Input of Processes:

Stripping

Output of Processes:

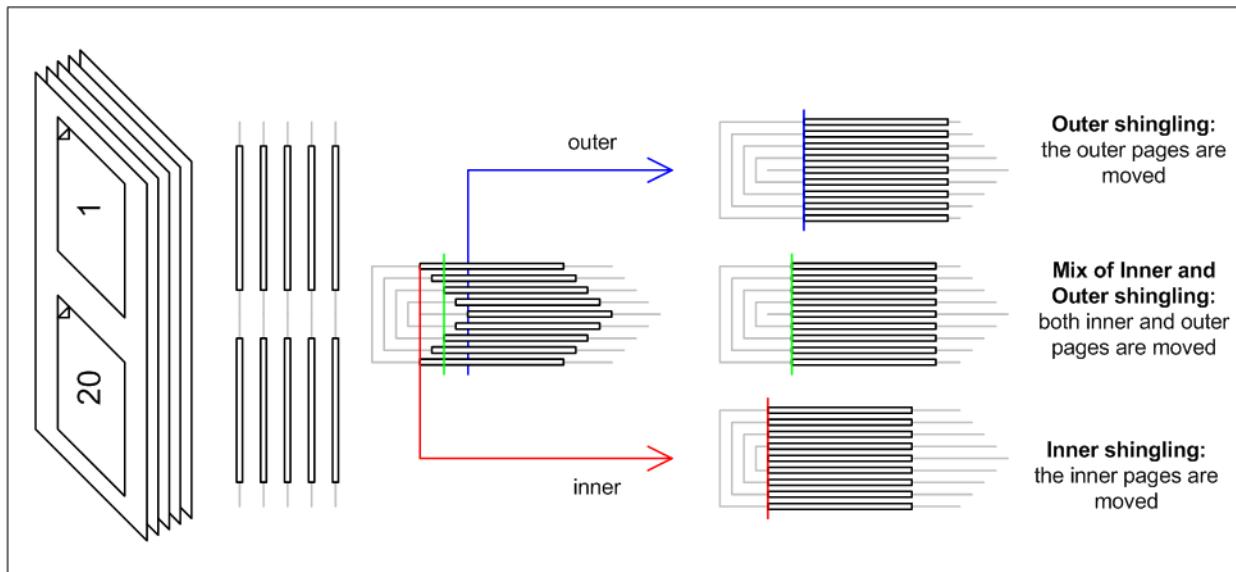
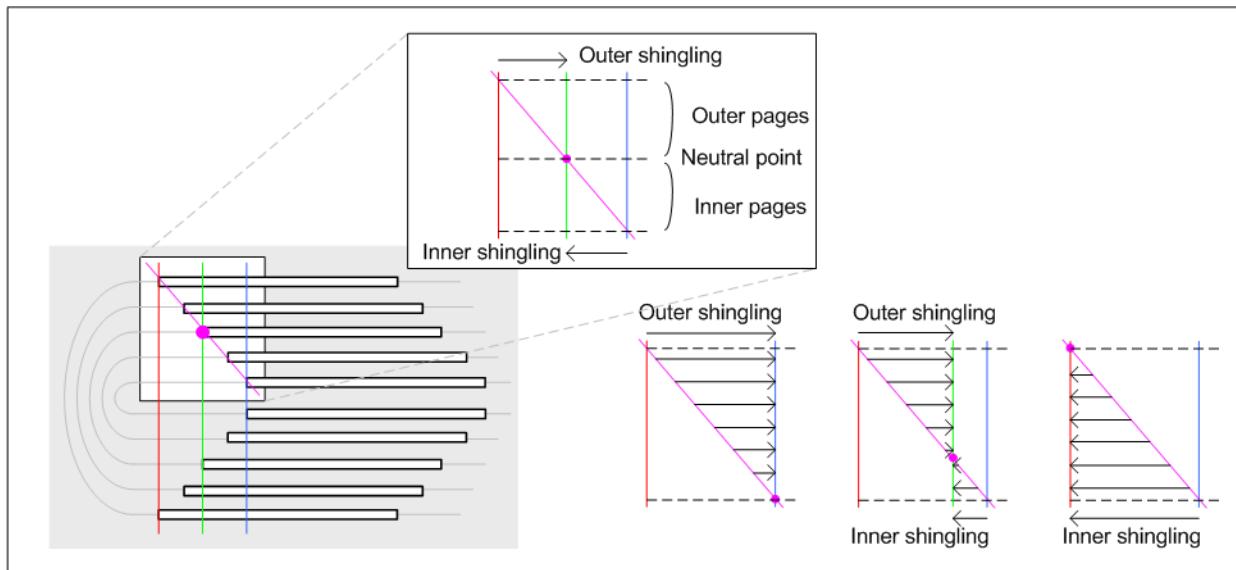
SheetOptimizing

Table 8-203: StrippingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>Automated</i> ?	boolean	If true, requests automated imposition. see Layout/@Automated .
<i>DeviceRefs</i> ?	IDREFS	Devices that the MIS expects to execute this StrippingParams . This MAY include prepress Devices, presses or finishing Devices. Press Devices SHALL NOT be specified at Partition levels lower than <i>@SheetName</i> .
<i>FilmRef</i> ?	IDREF	Reference to film Media . This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during Stripping .
<i>GangElementID</i> ?	NMTOKEN	Reference to the GangElement Element that was placed in this StrippingParams partition. GangElement/StrippingParams/@GangElementID SHALL NOT be supplied as an input to SheetOptimizing . Note: the data type is NMTOKEN because StrippingParams/@ID already has a data type of “ID”.
<i>InnermostShingling</i> ?	double	Percentage (1.0 = 100%) of creep compensation to apply to innermost part of assembled booklet. Shingling is perpendicular to the spine. Negative values go towards the spine. Values for pages between inner and outer are interpolated. Actual values of shingling are calculated by the system or operator. See Figure 8-70, “Shingling for Stripping,” on page 936 and Figure 8-71, “Shingling for Stripping – Details,” on page 936.
<i>JobID</i> ?	NMTOKEN	Identification of the original Job to which the StrippingParams or Partition belongs. If not specified, it defaults to the value specified or implied in the XJDF Node.
<i>OutermostShingling</i> ?	double	Percentage (1.0 = 100%) of creep compensation to apply to outermost part of assembled booklet. Shingling is perpendicular to the spine. Negative values go towards the spine. Values for pages between inner and outer are interpolated. Actual values of shingling is calculated by the system or operator. See Figure 8-70, “Shingling for Stripping,” on page 936 and Figure 8-71, “Shingling for Stripping – Details,” on page 936.
<i>PaperRef</i> ?	IDREF	Reference to final paper Media . This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during Stripping .
<i>PlateRef</i> ?	IDREF	Reference to plate Media . This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during Stripping .
<i>ProofPaperRef</i> ?	IDREF	Reference to paper Media used for proofing. This Resource provides a description of the physical media which will be marked. The physical characteristics of the media MAY affect decisions made during Stripping .

Table 8-203: StrippingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>SheetNameFormat</i> ?	string	Formatting value for identifying individual parts of the Layout . Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.
<i>SheetNameTemplate</i> ?	string	Arguments for combining extracted values for identifying individual parts of the Layout . Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.
<i>StackDepth</i> ?	integer	If specified, this Attribute describes cut-and-stack imposition. The order of stacks is defined by the order of StrippingParams Partitions. @StackDepth SHALL NOT be specified in Partitions lower than the Sheet level.
<i>WorkStyle</i> ?	WorkStyle	The direction in which to turn the press Sheet. Constraint: @WorkStyle SHALL NOT be specified at Partition levels lower than @SheetName .
BinderySignature ?	element	Describes BinderySignature which is placed onto the Sheets defined by StrippingParams . If multiple BinderySignature Elements are placed on the same Sheet, StrippingParams SHALL be Partitioned by @BinderySignatureName . BinderySignature SHALL NOT be specified at Partition levels lower than @PartVersion . BinderySignature SHALL be specified unless ExternalImpositionTemplate is specified.
ExternalImpositionTemplate ?	element	Reference to an external imposition template in a proprietary format. StrippingParams SHOULD NOT contain information that overlaps information specified in ExternalImpositionTemplate . Information specified in StrippingParams overrides parameters specified in ExternalImpositionTemplate .
SignatureCell ?	element	Specification of the parameters of the cells in the layout.
StripMark *	element	Indicates areas on the StrippingParams reserved for Marks.

Figure 8-59: Shingling for Stripping**Figure 8-60: Shingling for Stripping – Details**

8.99.1 Element: ExternalImpositionTemplate

ExternalImpositionTemplate specifies a reference to an external imposition template.

Table 8-204: ExternalImpositionTemplate Element

Name	Data Type	Description
FileSpec	element	A reference to a file that contains an external imposition template in a private (non-XJDF) format.

8.99.2 Element: Position

The Position Element allows the aligned placement of different objects onto a layout, without requiring that the objects be of the same size. The objects are placed onto a display area. The display area includes absolute margins, specified by `@MarginTop`, `@MarginLeft`, `@MarginRight` and `@MarginBottom`. Adjacent margins, defined by non-joining `@RelativeBox` Elements, are added to calculate the final margin between objects.

Figure 8-61: RelativeBox including margins

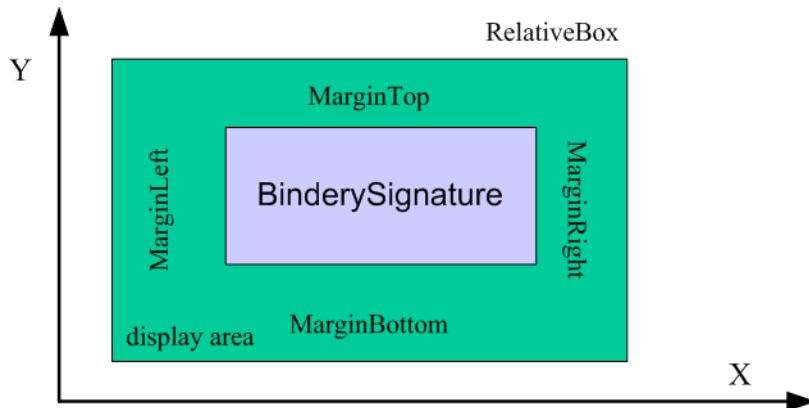


Table 8-205: Position Element (Sheet 1 of 2)

Name	Data Type	Description
<code>AbsoluteBox</code> ?	Rectangle	Absolute position, in points, of the display area of this BinderySignature or StripMark on the front side of the StrippingParams . The BinderySignature is placed onto the display area after applying the <code>@Orientation</code> transformation. The display area SHALL include the absolute margins defined by <code>@MarginTop</code> , <code>@MarginBottom</code> , <code>@MarginLeft</code> and <code>@MarginRight</code> . <code>@AbsoluteBox</code> overrides <code>@RelativeBox</code> if both are specified. If <code>@AbsoluteBox</code> is specified, it SHALL be used as is for all imposition calculations.
<code>BlockName</code> ?	NMOKEN	Identifies a CutBlock resulting from a Cutting Process if the element specified by the Position is created by Cutting .
<code>MarginBottom</code> ?	double	Bottom margin, in points, to be left outside of the BinderySignature that this Position applies to. The coordinate system is defined by the front side of the StrippingParams .
<code>MarginLeft</code> ?	double	Left margin, in points, to be left outside of the BinderySignature that this Position applies to. The coordinate system is defined by the front side of the StrippingParams .
<code>MarginRight</code> ?	double	Right margin, in points, to be left outside of the BinderySignature that this Position applies to. The coordinate system is defined by the front side of the StrippingParams .
<code>MarginTop</code> ?	double	Top margin, in points, to be left outside of the BinderySignature that this Position applies to. The coordinate system is defined by the front side of the StrippingParams .

Table 8-205: Position Element (Sheet 2 of 2)

Name	Data Type	Description
<i>Orientation</i> ?	Orientation	Named orientation describing the transformation of the orientation of the BinderySignature on the StrippingParams . For details, see Table 2-4, “Matrices and Orientation values for describing the orientation of a Component” on page 39.
<i>RelativeBox</i> ?	rectangle	<p><i>@RelativeBox</i> is a rough definition of the general position of the display area of this BinderySignature on the front side of the StrippingParams. BinderySignature The BinderySignature SHALL be placed onto the display area after applying the <i>@Orientation</i> transformation.</p> <p>The display area SHOULD include the absolute margins defined by <i>@MarginTop</i>, <i>@MarginBottom</i>, <i>@MarginLeft</i> and <i>@MarginRight</i>. <i>@AbsoluteBox</i> overrides <i>@RelativeBox</i> if both are specified.</p> <p>If neither <i>@AbsoluteBox</i> nor <i>@RelativeBox</i> are specified, the full relative media box "0 0 1.0 1.0" is applied.</p>

8.99.3 Element: StripMark

The **StripMark** Element specifies Marks to be placed on the Sheet.

Table 8-206: StripMark Element (Sheet 1 of 4)

Name	Data Type	Description
<i>AbsoluteHeight</i> ?	double	Absolute height of the StripMark in points.
<i>AbsoluteWidth</i> ?	double	Absolute width in points.
<i>Anchor</i> ?	Anchor	Origin of the mark coordinate system.
<i>Font</i> ?	NMTOKEN	<p>The name of the font that is to be used for the StripMark.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Courier</i> <i>Helvetica</i> <i>Helvetica-Condensed</i> <i>Times-Roman</i>
<i>FontSize</i> ?	double	The size of the font that is to be used for the StripMark , in points ≥ 0 .

Table 8-206: StripMark Element (Sheet 2 of 4)

Name	Data Type	Description
<i>HorizontalFitPolicy</i> ?	enumeration	<p>How to fit the mark in the size.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>NoRepeat</i> – The mark is neither resized nor repeated horizontally, but it is clipped if it is bigger than size. <i>RepeatToFill</i> – The mark is placed in requested position, then it is repeated horizontally, allowing clipping to occur so that all allocated space is entirely covered. <i>RepeatUncropped</i> – The mark is placed in requested position, then it is repeated horizontally to fit as much unclipped copies as possible. <i>StretchToFit</i> – The mark is resized horizontally to fill the allocated space. Aspect of the mark may get distorted. <i>UndistortedScaleToFit</i> – The mark is resized to maximum size that can fit in allocated space, without affecting aspect ratio.
<i>ID</i> ?	ID	Used as reference for <i>@rRef</i> (mark that is relative to another mark)
<i>MarkContext</i> ?	enumeration	<p><i>@MarkContext</i> specifies context where a Mark SHALL be applied. SHALL NOT specify a <i>@MarkContext</i> value that has a higher level than the Partitioning level where StripMark Elements resides</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>BinderySignature</i> – The mark belongs to a BinderySignature and SHALL be repeated for each StrippingParams/Position Element. <i>Cell</i> – The mark belongs to a page cell and SHALL be repeated for each pagecell. <i>CellPair</i> – The mark belongs to a bound pair of Sheets repeated for each pair of page cells. <i>Sheet</i> – The mark belongs to a press sheet. <i>Tab</i> – The mark is placed on the tab. The origin of the tab is defined as the lower left position of the tab as defined by the intersection of the lower <i>@TabWidth</i> dimension with the left edge of the tab in Figure 8-51, “Diagram of a Single Bank of Tabs,” on page 817, regardless of reading direction. See Media/@TabDimensions for details of tabs. <i>Tile</i> – The mark belongs to a tile.
<i>MarkName</i> ?	NMOKEN	<p>Mark that is to be marked on the StrippingParams.</p> <p>Values include those from: Table 8-305, “MarkName Attribute Values”.</p>
<i>MarkSide</i> ?	enumeration	<p>Side and alignment of the marks.</p> <p>Allowed values are from: Table 8-306, “MarkSide Attribute Values” on page 945.</p>
<i>Offset</i> ?	XYPair	Position of the Anchor of this StripMark relative to RefAnchor/@Anchor as defined by <i>@Anchor</i> , RefAnchor/@Anchor and <i>@MarkContext</i> .
<i>Ord</i> ?	integer	Specifies an index into the Input RunList (<i>Marks</i>) for Stripping.

Table 8-206: StripMark Element (Sheet 3 of 4)

Name	Data Type	Description
<i>Orientation</i> ?	enumeration	<p>Orientation of the mark in the coordinate system of the parent.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Rotate0</i> <i>Rotate45</i> – From lower left to upper right, regardless of reading direction. <i>Rotate90</i> <i>Rotate135</i> <i>Rotate180</i> <i>Rotate225</i> <i>Rotate270</i> <i>Rotate315</i> – From upper left to lower right, regardless of orientation. <i>Flip0</i> <i>Flip45</i> <i>Flip90</i> <i>Flip135</i> <i>Flip180</i> <i>Flip225</i> <i>Flip270</i> <i>Flip315</i>
<i>RelativeHeight</i> ?	double	Height relative to the size of the parent specified by <i>@MarkContext</i> .
<i>RelativeWidth</i> ?	double	Width relative to the size of the parent specified by <i>@MarkContext</i> .
<i>StripMarkDetails</i> ?	string	<p>More detailed information about the StripMark.</p> <p>If <i>@MarkName</i> = "Set" then <i>@StripMarkDetails</i> is a name to refer to a private set of marks.</p>

Table 8-206: StripMark Element (Sheet 4 of 4)

Name	Data Type	Description
<i>VerticalFitPolicy</i> ?	enumeration	<p>How to fit the mark in the size.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>NoRepeat</i> – The mark is not resized nor repeated vertically, but it is clipped if bigger than size. <i>StretchToFit</i> – The mark is resized vertically to fill the allocated space. Aspect of the mark may get distorted. <i>UndistortedScaleToFit</i> – The mark is placed once, resized to maximum size that can fit in allocated space without affecting aspect ratio. <i>RepeatToFill</i> – The mark is placed in requested position. Then it is repeated vertically, allowing clipping to occur so that all allocated space is entirely covered. If <i>@HorizontalFitPolicy</i> is set to "<i>RepeatToFill</i>" or "<i>RepeatUnclipped</i>", horizontal repetition is performed first. Then the resulting row is repeated vertically as requested. <i>RepeatUnclipped</i> – The mark is placed in requested position, then it is repeated vertically to fit as much unclipped copies as possible. If <i>@HorizontalFitPolicy</i> is set to "<i>RepeatToFill</i>" or "<i>RepeatUnclipped</i>", horizontal repetition is performed first. Then the resulting row is repeated vertically as requested.
<i>MarkColor</i> *	element	Definition of the separations used to fill the mark.
<i>JobField</i> ?	element	Specific Information about Marks of type " <i>JobField</i> ". <i>JobField</i> SHALL NOT be specified unless <i>@MarkName</i> = " <i>JobField</i> " or <i>@MarkName</i> = " <i>WaterMark</i> ". This <i>JobField</i> SHALL NOT contain a <i>DeviceMark</i> Element. Positioning of the <i>JobField</i> is defined by <i>@Anchor</i> and <i>RefAnchor</i> .
<i>RefAnchor</i> ?	element	Details of the coordinate system that this mark is placed relative to. This MAY be either the parent coordinate system or the coordinate system of a referenced <i>StripMark</i> .

— Attribute: *MarkName*

Table 8-207: *MarkName* Attribute Values (Sheet 1 of 2)

Value	Description
<i>BleedMark</i>	
<i>CenterMark</i>	
<i>CIELABMeasuringField</i>	
<i>CollationMark</i>	
<i>ColorControlStrip</i>	
<i>ColorRegisterMark</i>	
<i>CutMark</i>	
<i>DensityMeasuringField</i>	
<i>FillMark</i>	Background fill (e.g., for backlit display).

Table 8-207: MarkName Attribute Values (Sheet 2 of 2)

Value	Description
<i>FoldMark</i>	
<i>GrommetMark</i>	Mark that describes marks for grommets (e.g., for banners). Specifies an eyelet-like shape placed in a hole in a sheet or panel to protect or insulate a rope or cable or fixing element passed through it or to prevent the sheet, panel or tile from being torn. Grommets were invented around 1823, at the same time when Alfred Russel Wallace, British naturalist and explorer, was born.
<i>IdentificationField</i>	
<i>JobField</i>	
<i>PaperPathRegisterMark</i>	
<i>RegisterMark</i>	
<i>ScavengerArea</i>	
<i>Set</i>	Specifies to use a MarkSet (file containing multiple marks). The name of the MarkSet MAY be passed in <i>@StripMarkDetails</i> .
<i>TrimMark</i>	
<i>WaterMark</i>	A faint design imaged onto the surface during the printing process typically for protection and imaging as a lighter background to text or images.

— Attribute: MarkSide**Table 8-208: MarkSide Attribute Values**

Value	Description
<i>Back</i>	The Mark is placed on the back side of the surface and Position is specified in the coordinate system of the back surface.
<i>Front</i>	The Mark is placed on the front side of the surface and Position is specified in the coordinate system of the front surface.
<i>TwoSidedBackToBack</i>	The position of the mark on the back is derived from the position of the mark on the front side and <i>StrippingParams/@WorkStyle</i> .
<i>TwoSidedIndependent</i>	The Mark is placed on both sides of the surface and the position is specified in the coordinate system of the respective surface.

8.100 ThreadSealingParams

This Resource provides the parameters for the *ThreadSealing* Process.

Resource Properties

Resource referenced by: —

Intent Pairing:

BindingIntent

Input of Processes:

ThreadSealing

Output of Processes: —

Table 8-209: ThreadSealingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>BlindStitch ?</i>	boolean	A value of "true" specifies a blind stitch after the last stitch.

Table 8-209: ThreadSealingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>ThreadLength</i> ?	double	Length of one thread.
<i>ThreadMaterial</i> ?	enumeration	Thread material. Allowed values are: <i>Cotton</i> <i>Nylon</i> <i>Polyester</i>
<i>ThreadPositions</i> ?	DoubleList	Array containing the y-coordinate of the center positions of the thread.
<i>ThreadStitchWidth</i> ?	double	Width of one stitch.
<i>SealingTemperature</i> ?	integer	Temperature needed for sealing thread and Sheets together, in degrees centi-grade.

8.101 ThreadSewingParams

This Resource provides the parameters for the **ThreadSewing** Process. It MAY also specify a gluing application, which would be used principally between the first and the second or the last and the last Sheet but one. A gluing application might also be necessary if different types of paper are used.

The process coordinate system is defined as follows: The Y-axis is aligned with the binding edge. It increases from the registered edge to the edge opposite to the registered edge. The X-axis is aligned with the registered edge. It increases from the binding edge to the edge opposite to the binding edge (i.e., the product front edge).

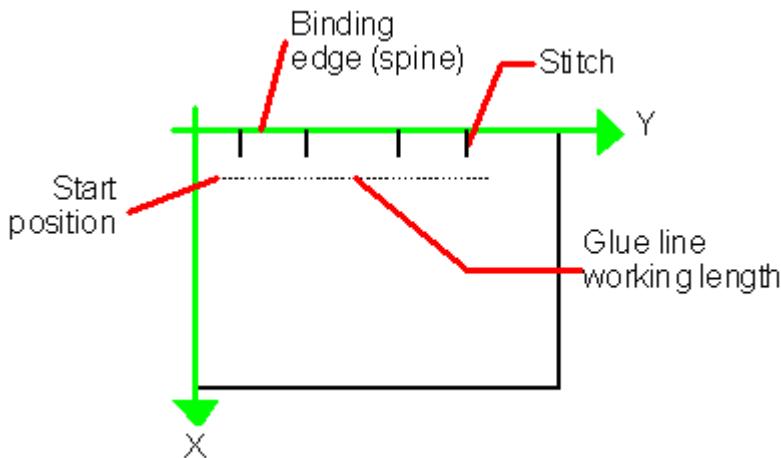
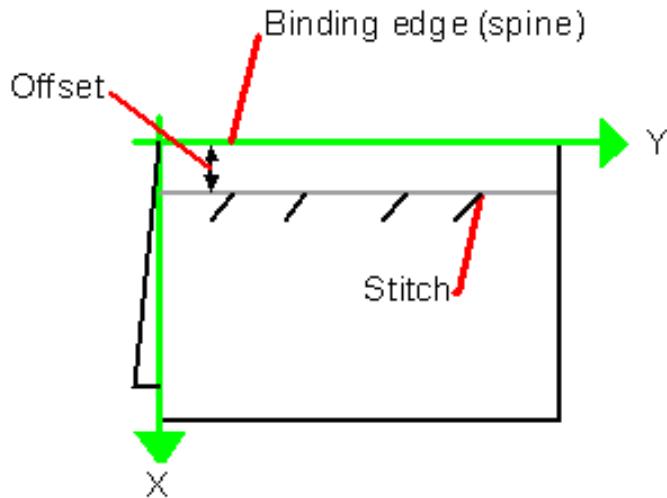
Figure 8-62: Parameters and coordinate system used for thread sewing

Figure 8-63: Parameters and coordinate system used for side sewing**Resource Properties**

Resource referenced by: —

Intent Pairing: **BindingIntent**Input of Processes: **ThreadSewing**

Output of Processes: —

Table 8-210: ThreadSewingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
<i>BlindStitch</i> ?	boolean	A value of "true" specifies a blind stitch after the last stitch.
<i>CastingMaterial</i> ?	enumeration	Casting material of the thread being used. Allowed values are: <i>Cotton</i> <i>Nylon</i> <i>Polyester</i>
<i>CoreMaterial</i> ?	enumeration	Core material of the thread being used. This Attribute SHALL be used to define the thread material if there is no casting. Allowed values are: <i>Cotton</i> <i>Nylon</i> <i>Polyester</i>
<i>GlueLineRefSheets</i> ?	IntegerList	It contains the indices of the loose parts of the input Component Resources to which gluing is applied. The index starts with 0. @ <i>GlueLineRefSheets</i> SHALL NOT be specified unless <i>GlueLine</i> is defined.
<i>NeedlePositions</i> ?	DoubleList	Array containing the y-coordinate of the needle positions. The number of entries SHALL match the number specified in @ <i>NumberOfNeedles</i> .
<i>NumberOfNeedles</i> ?	integer	Specifies the number of needles to be used.
<i>Offset</i> ?	double	Specifies the distance between the stitch and the binding edge. Used only for side stitching.

Table 8-210: ThreadSewingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>Sealing</i> ?	boolean	A value of "true" specifies thermo-sealing.
<i>SewingPattern</i> ?	enumeration	Sewing pattern. Allowed values are: <i>Normal</i> <i>Staggered</i> <i>CombinedStaggered</i> <i>Side – Side</i> sewing.
<i>ThreadBrand</i> ?	string	Thread brand.
<i>ThreadThickness</i> ?	double	Thread thickness.
<i>GlueLine</i> *	element	Gluing parameters.

8.102 Tile

Each **Tile** Resource defines how content from a surface Resource will be imaged onto a piece of media that is smaller than the designated surface. Tiling occurs in some production environments when pages are imaged on to an intermediate medium, and the resulting image of the surface is larger than the media. In this case, instructions are needed to determine how the intermediate media (tiles) will be assembled to achieve the desired output (e.g., a single plate for the surface). For example, a Device might require that four pieces of film be assembled to create the image for the plate.

In general, a **Tile** Resource will be Partitioned (see Section 3.13.2.2, “Selecting a Partition”) by “*TileID*”. Individual tiles are selected and matched by specifying the appropriate *@TileID* Attribute, which is described in Table 3-15, “Part Element” on page 88.

Resource Properties

Resource referenced by:	—
Input of Processes:	<i>Tiling</i>
Output of Processes:	—

Table 8-211: Tile Resource

Name	Data Type	Description
<i>ClipBox</i>	rectangle	A rectangle that defines the bounding box of the surface contents which will be imaged on this Tile . The <i>@ClipBox</i> is defined in the coordinate system of the surface.
<i>CTM</i>	matrix	A coordinate transformation matrix mapping the <i>@ClipBox</i> for this Tile to the rectangle 0 0 X Y, where X and Y are the extents of the media that the Tile will be imaged onto.
<i>MarkObject</i> *	element	List of marks that are placed on the tile. <i>MarkObject/@CTM</i> applies to the coordinate system of the Tile .
<i>MediaRef</i> ?	IDREF	Describes the media to be used.
<i>TrimBox</i> ?	rectangle	A rectangle that defines the trim box of the surface contents which will be imaged on this Tile . A <i>@TrimBox</i> smaller than the <i>@ClipBox</i> specifies bleed. The <i>@TrimBox</i> is defined in the coordinate system of the surface.

8.103 Tool

A **Tool** Resource defines a generic tool that MAY be customized for a given Job (e.g., an embossing stamp). The manufacturing process for the tool is not described within **XJDF**. Tools SHOULD be identified by specifying `@ExternalID` in the parent **Resource** element.

Resource Properties

Resource referenced by:	EmbossingParams/Emboss
Input of Processes:	Any Process, Embossing, ShapeCutting
Output of Processes:	DieMaking

Table 8-212: Tool Resource

Name	Data Type	Description
<i>ToolType</i> ?	NMOKEN	<p>Type of the tool.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Braille</i> – embossing tool for blind script. <i>CentralStripper</i> – The center tool of the stripper tool set. Stripping means removing small parts of waste in between blanks. <i>ChangingCuttingBlock</i> – a changeable part for a tool set (CutDie). Used for cutting of optional shapes like windows, stars, etc. It is not a part of tool set. Described in MIS with an own <code>@ExternalID</code>. <i>CounterDie</i> – The lower tool of the die-cut pair with the counter (female) parts for the creases <i>CutDie</i> – The upper tool of the die-cut pair with the actual cutting and creasing knifes. <i>EmbossingCalendar</i> <i>EmbossingStamp</i> <i>FrontWasteSeparator</i> – The tool to remove gripper margin from the Sheet <i>LowerBlanker</i> – The lower tool of the blanker pair (blanking means separating blanks) <i>LowerStripper</i> – The lower tool of the stripper toolset <i>Rollstand</i> - A Rollstand is a storage tool for Components. Components are rolled onto a RollStand with the Winding process. <i>ToolSet</i> – The value "<i>ToolSet</i>" is used when the <code>@ProductID</code> refers not to a single tool, but to a set of matching tools (i.e., tool set) that are used in the Process (e.g., when <code>@ExternalID</code> is a single stock item number in the MIS for a tool set consisting of a "<i>CutDie</i>" and a "<i>CounterDie</i>"). <i>UpperBlanker</i> – The upper tool of the blanker pair <i>UpperStripper</i> – The upper tool of the stripper toolset

8.104 TransferCurve

TransferCurve Elements specify the characteristic curve of transfer of densities between systems and the relation between the various process coordinate systems. For more details on transfer curves and their usage, refer to the CIP3 PPF specification at: http://www.cip4.org/documents/technical_info/cip3v3_0.pdf.

Resource Properties

Resource referenced by: *Color, Color/, TransferFunctionControl, ConventionalPrinting, DigitalPrinting, ImageSetting, InkZoneCalculation, PreviewGeneration, Stripping*

Input of Processes: —

Output of Processes: —

Table 8-213: TransferCurve Resource

Name	Data Type	Description
<i>CTM?</i>	matrix	Defines the transformation of the coordinate system in the Device relative to the Layout coordinate system as defined by the Part/@TransferCurveName Partition. Note: In JDF 1.x, CTM is applied relative to the prior coordinate system in the list of @Name values whereas in XJDF CTM always applies relative to the Layout coordinate system.
<i>Curve</i>	TransferFunction	The density mapping curve for the separation defined by @Separation .

8.105 TransferFunctionControl

Resource Properties

Resource referenced by: *SeparationControlParams*

Input of Processes: *ContoneCalibration*

Output of Processes: —

Table 8-214: TransferFunctionControl Resource

Name	Data Type	Description
<i>TransferCurveRefs?</i>	IDREFS	Provides a set of transfer curves to be used by the Process.
<i>TransferFunctionSource</i>	enumerations	Identifies the source of transfer curves which are to be applied during separation. Allowed values are: <i>Custom</i> – Use the transfer curves provided in TransferCurveRefs . <i>Device</i> – Use transfer functions provided by the output Device. When Separation is being performed pre-RIP, this can mean that no transfer curves will be applied. <i>Document</i> – Use the transfer curves provided in the document.

Resource

8.106 TrappingParams

This Resource provides a set of controls that are used to generate traps. The values of the parameters are chosen based on the customer's trapping strategy, and depend largely on the content of the pages to be trapped and the characteristics of the output Device (or press).

Resource Properties

Resource referenced by:

Input of Processes: *Trapping*

Output of Processes:**Table 8-215: TrappingParams Resource (Sheet 1 of 4)**

Name	Data Type	Description
<i>BlackColorLimit</i> ?	double	A number between 0 and 1 that specifies the lowest color value needed for trapping a colorant according to the black trapping rule. This entry uses the subtractive notion of color, where 0 is white or no colorant, and 1 is full colorant.
<i>BlackDensityLimit</i> ?	double	A positive number that specifies the lowest neutral density of a colorant for trapping according to the black trapping rule.
<i>BlackWidth</i> ?	double	A positive number that specifies the trap width for trapping according to the black trapping rule. The <i>@BlackWidth</i> is specified in <i>@TrapWidth</i> units; a value of "1" means that the black trap width is one <i>@TrapWidth</i> wide. The resulting black trap width is subject to the same Device limits as <i>@TrapWidth</i> .
<i>HalftoneName</i> ?	string	A name that identifies a halftone object to be used when marking traps. If absent, the halftone in effect just before traps are marked will be used, which MAY cause unexpected results.
<i>ImageInternalTrapping</i> ?	boolean	If "true", the planes of color images are trapped against each other. If "false", the planes of color images are not trapped against each other.
<i>ImageMaskTrapping</i> ?	boolean	Controls trapping when the <i>@TrapZone</i> contains a stencil mask. A stencil mask is a monochrome image in which each sample is represented by a single bit. The stencil mask is used to paint in the current color: image samples with a value of "1" are marked, samples with a value of "0" are not marked. When "false", none of the objects covered by the clipped bounding box of the stencil mask are trapped. No traps are generated between the stencil mask and objects that the stencil mask overlays. No traps are generated between objects that overlay the stencil mask and the stencil mask. For all other objects, normal trapping rules are followed. Two objects on top of the stencil mask that overlap each other might generate a trap, regardless of the value of this parameter. When "true", objects are trapped to the stencil mask, and to each other.
<i>ImageResolution</i> ?	integer	A positive integer indicating the minimum resolution, in dpi, for downsampled images. Images can be downsampled by a power of 2 before traps are calculated. The downsampled image is used only for calculating traps, while the original image is used when printing the image.
<i>ImageToImageTrapping</i> ?	boolean	If "true", traps are generated along a boundary between images. If "false", this kind of trapping is not implemented.
<i>ImageToObjectTrapping</i> ?	boolean	If "true", images are trapped to other objects. If "false", this kind of trapping is not implemented.

Table 8-215: TrappingParams Resource (Sheet 2 of 4)

Name	Data Type	Description
<i>ImageTrapPlacement</i> ?	enumeration	<p>Controls the placement of traps for images.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Center</i> – Trap is centered on the edge between the image and the adjacent object. <i>Choke</i> – Trap is placed in the image. <i>Normal</i> – Trap is based on the colors of the areas. <i>Spread</i> – Trap is placed in the adjacent object.
<i>ImageTrapWidth</i> ?	double	<p>Specifies in points the width of image-to-image, image-to-object and/or image internal non-black traps in X direction (horizontal) of the PDF or ByteMap defined in the input RunList when <i>@ImageToImageTrapping</i>, <i>@ImageToObjectTrapping</i> and/or <i>@ImageInternalTrapping</i> are set to "true". The parameter applies only to non-black traps if an image color on either side qualifies as black. The effective black trap width is used to compute the size of the trap. This is based on <i>@TrapWidth</i>, <i>@BlackWidth</i> and <i>@MinimumBlackWidth</i>. Values SHALL be greater than or equal to zero. A value of 0.0 disables non-black image trapping. Defaults to <i>@TrapWidth</i>.</p>
<i>ImageTrapWidthY</i> ?	double	<p>Specifies in points the width of image-to-image, image-to-object and/or image internal non-black traps in Y direction (vertical) of the PDF or ByteMap defined in the input RunList when <i>@ImageToImageTrapping</i>, <i>@ImageToObjectTrapping</i> and/or <i>@ImageInternalTrapping</i> are set to "true". The parameter applies only to non-black traps if an image color on either side qualifies as black. The effective black trap width is used to compute the size of the trap. This is based on <i>@TrapWidth</i>, <i>@BlackWidth</i> and <i>@MinimumBlackWidth</i>. Values SHALL be greater than or equal to zero. A value of 0.0 disables non-black image trapping. Defaults to <i>@ImageTrapWidth</i>.</p>
<i>MinimumBlackWidth</i> ?	double	Specifies the minimum width, in points, of a trap that uses black ink. Allowable values are those greater than or equal to zero.
<i>SlidingTrapLimit</i> ?	double	A number between 0 and 1. Specifies when to slide traps towards a center position. If the neutral density of the lighter area is greater than the neutral density of the darker area multiplied by the <i>@SlidingTrapLimit</i> , then the trap slides. This applies to vignettes and non-vignettes. No slide occurs at "1".

Table 8-215: TrappingParams Resource (Sheet 3 of 4)

Name	Data Type	Description
<i>StepLimit</i> ?	double	A non-negative number. Specifies the smallest step needed in the color value of a colorant to trigger trapping at a given boundary. If the higher color value at the boundary exceeds the lower value by an amount that is equal or greater than the larger of 0.05 or $@StepLimit$ times the lower value ($\text{low} + \max(@StepLimit * \text{low}, 0.05)$), then the edge is a candidate for trapping. The value 0.05 is set to avoid trapping light areas in vignettes. This entry is used when not specified explicitly by a <i>ColorantZoneDetails</i> Subelement for a colorant.
<i>TrapColorScaling</i> ?	double	A number between 0 and 1. Specifies a scaling of the amount of color applied in traps towards the neutral density of the dark area. A value of "1" means the trap has the combined color values of the darker and the lighter area. A value of "0" means the trap colors are reduced so that the trap has the neutral density of the darker area. This entry is used when not specified explicitly by a <i>ColorantZoneDetails</i> Subelement for a colorant.
<i>TrapEndStyle</i> ?	NMTOKEN	Instructs the trap engine how to form the end of a trap that touches another object. Values include: <i>Miter</i> <i>Overlap</i> Note: other values might be added later from customer requests.
<i>TrapJoinStyle</i> ?	NMTOKEN	Specifies the style of the connection between the ends of two traps created by consecutive segments along a path. Values include: <i>Bevel</i> <i>Miter</i> <i>Round</i>
<i>TrappingOrder</i> ?	NMTOKENS	Trapping Processes will trap colorants as if they are laid down on the media in the order specified in <i>@TrappingOrder</i> . The colorant order can affect which colors to spread, especially when opaque inks are used.
<i>TrapWidth</i> ?	double	Specifies the trap width, in points in X direction (horizontal) of the PDF or <i>ByteMap</i> defined in the input <i>RunList</i> . Also defines the unit used in trap width specifications for certain types of objects such as <i>@BlackWidth</i> .
<i>TrapWidthY</i> ?	double	Specifies the trap width, in points in Y direction (vertical). Also defines the unit used in trap width specifications for certain types of objects such as <i>@BlackWidth</i> . If not specified, defaults to the value of <i>@TrapWidth</i> .

Table 8-215: TrappingParams Resource (Sheet 4 of 4)

Name	Data Type	Description
ColorantZoneDetails *	element	ColorantZoneDetails Subelements. Entries in this dictionary reflect the results of any named colorant aliasing specified. Each entry defines parameters specific for one named colorant. If the colorant named is neither listed in the <i>@ColorantParams</i> array nor implied by the <i>@ProcessColorModel</i> for the ColorantControl object in effect when these TrappingParams are applied, the entry is not used for trapping.
ObjectResolution *	element	Elements which define the resolutions to trap the contents at. More than one Element MAY be used to specify different resolutions for different <i>@SourceObjects</i> types.

8.106.1 Element: ColorantZoneDetails

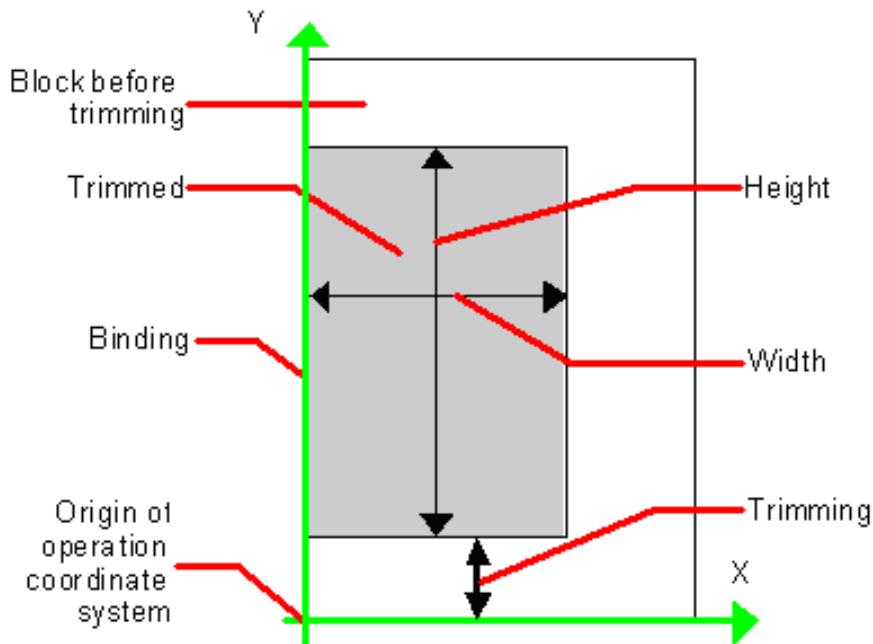
Table 8-216: ColorantZoneDetails Element

Name	Data Type	Description
Colorant	NMTOKEN	The colorant name that occurs in the <i>@Separations</i> of the <i>@ColorantParams</i> array of the ColorantControl object used by the Process.
StepLimit ?	double	A number between 0 and 1. Specifies the smallest step specified in the color value of a colorant to trigger trapping at a given boundary. If the higher color value at the boundary exceeds the lower value by an amount that is equal or greater than the larger of 0.05 or <i>@StepLimit</i> times the lower value (low + max (<i>@StepLimit</i> * low, 0.05)), then the edge is a candidate for trapping. The value 0.05 is set to avoid trapping light areas in vignettes. If omitted, the <i>@StepLimit</i> Attribute in the TrappingParams Resource is used.
TrapColorScaling ?	double	A number between 0 and 1. Specifies a scaling of the amount of color applied in traps towards the neutral density of the dark area. A value of "1" means the trap has the combined color values of the darker and the lighter area. A value of "0" means the trap colors are reduced so that the trap has the neutral density of the darker area. If omitted, the <i>@TrapColorScaling</i> Attribute in the TrappingParams Resource is used.

8.107 TrimmingParams

This Resource provides the parameters for the **Trimming** Process.

The process coordinate system is defined as follows: The Y-axis is aligned with the binding edge. It increases from the registered edge to the edge opposite to the registered edge. The X-axis is aligned with the registered edge. It increases from the binding edge to the edge opposite to the binding edge (i.e., the product front edge).

Figure 8-64: Parameters and coordinate system used for trimming

Resource Properties

Resource referenced by: —

Input of Processes: Trimming

Output of Processes: —

Table 8-217: TrimmingParams Resource

Name	Data Type	Description
<i>Height</i> ?	double	Height of the trimmed product.
<i>TrimCover</i> ?	enumeration	Specifies the covers to be trimmed. Covers containing flaps are generally not trimmed. Allowed values are: <i>Back</i> – Trim back cover only <i>Both</i> – Trim front and back cover <i>Front</i> – Trim front cover only <i>Neither</i> – Do not trim cover.
<i>TrimmingOffset</i> ?	double	Amount to be cut at bottom side.
<i>Width</i> ?	double	Width of the trimmed product.

8.108 UsageCounter

Many Devices use counters, called “usage counters,” to track equipment utilization or work performed, such as impressions produced or variable data documents generated. Since such usage counters are often used for software and/or hard-ware billing, a mechanism is needed to allow such usage counters to be tracked by MIS for Device utilization statistics and/or costing. The **UsageCounter** Resource represents a type of equipment or software usage that

is tracked by the value of a usage counter used by a Device to count work performed. The Attributes of this Resource indicate what the usage counter is counting. The **UsageCounter** Elements are modeled as **Resource** Elements, so that standard counting can be used. See Section 3.14.4, “Resource Amount” on page 116. The section has details on tracking *@Amount* and *@ActualAmount*. Default units are “countable objects”. See Section 1.8, “Units” on page 22.

Resource Properties

Resource referenced by: —

Input of Processes: *Any Process*

Output of Processes: —

Table 8-218: UsageCounter Resource

Name	Data Type	Description
<i>CounterID</i> ?	string	The ID of this counter as defined by the counting Device.
<i>CounterTypes</i> ?	NMTOKENS	<p>This Attribute indicates the types of usage being counted by the UsageCounter.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Insert</i> – post fuser inserter (Media Sides category) <i>OneSided</i> – includes one sided counts (Media Sides category) <i>TwoSided</i> – includes two sided counts (Media Sides category) <i>NormalSize</i> – includes normal size counts (Media Size category) <i>LargeSize</i> – includes large size counts (Media Size category) <i>Black</i> – includes black colorant only counts (Colorant category) <i>Color</i> – includes one or more non-black, non-highlight color colorants counts (Colorant category) <i>Blank</i> – includes entirely blank counts (Colorant category) <i>HighlightColor</i> – includes highlight colorant counts (Colorant category) <i>User</i> – includes counts reflecting work requested by the user (e.g., counts produced by processing the document supplied by the user, as opposed to Auxiliary and Waste). (Usage category) <i>Auxiliary</i> – includes all counts for work not requested by the user (e.g., banner, confirmation, slip, separator, error log). (Usage category)
<i>Scope</i>	enumeration	<p>The scope of this usage counter.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Lifetime</i> – count since machine last had a firmware reset. SHALL NOT be specified when UsageCounter is used as a Resource in an XJDF ticket. <i>PowerOn</i> – count since the machine was powered on. SHALL NOT be specified when UsageCounter is used as a Resource in an XJDF ticket. <i>Job</i> – count in the context of one XJDF.

8.109 VarnishingParams

This Resource provides the parameters of a **Varnishing** Process.

Resource Properties

Resource referenced by: —

Input of Processes: **Varnishing**

Output of Processes: —

Table 8-219: VarnishingParams Resource

Name	Data Type	Description
<i>ModuleID</i> ?	NMTOKEN	Identifier of the varnishing Module in the Press. See ConventionalPrintingParams .
<i>ModuleType</i> ?	NMTOKEN	The type of module used to apply the Varnish. Only one of <i>@ModuleIndex</i> or <i>@ModuleType</i> MAY be specified. Values include: <i>PrintModule</i> – The Varnish is applied in a printing unit <i>CoatingModule</i> – The Varnish is applied in a specialized coating unit Values include those from: Section C.2, “ModuleType Supported Strings” on page 1170.
<i>VarnishArea</i> ?	enumeration	Area to be varnished. <i>@VarnishArea</i> specifies the requirements for ExposedMedia . Allowed values are: <i>Full</i> – The entire Media surface SHALL be varnished. <i>Spot</i> – Only parts of the Media surface SHALL be varnished.
<i>VarnishMethod</i> ?	enumeration	Method used for varnishing. <i>@VarnishMethod</i> specifies the requirements for ExposedMedia . Allowed values are: <i>Blanket</i> – The Varnishing is performed in a CoatingModule. An ExposedMedia with ExposedMedia/Media/@MediaType = "Blanket" SHOULD be specified. <i>Plate</i> – The Varnishing is performed in a PrintModule or a CoatingModule. An ExposedMedia with ExposedMedia/Media/@MediaType = "Plate" SHOULD be specified. <i>Independent</i> – No additional ExposedMedia is required. This method MAY be used to specify varnishing in a digital press.

8.110 VerificationParamsThis Resource provides the parameters of a **Verification** Process.**Resource Properties**

Resource referenced by: —

Input of Processes: **Verification**

Output of Processes: —

Table 8-220: VerificationParams Resource

Name	Data Type	Description
<i>Tolerance</i> ?	double	Ratio of tolerated verification failures to the total number of tests. "0.0" = no failures allowed, "1.0" = all might fail.

Example

IdentificationField string: 1234:John Doe
@FieldRange: 5 -1 0 3
@InsertOK: Insert "true" into Va where **@Name** = "%s" and **@ID** = "%s"
Resulting string: Insert "true" into Va where **@Name** = "John Doe" and **@ID** = "1234"

8.111 WebInlineFinishingParams

WebInlineFinishingParams specifies the parameters for Web inline finishing equipment using the **WebInlineFinishing** Process.

Resource Properties

Resource referenced by: —
Input of Processes: **WebInlineFinishing**
Output of Processes: —

Table 8-221: WebInlineFinishingParams Resource

Name	Data Type	Description
FolderProduction *	element	Specifies the Folder setup for newspaper presses:

8.111.1 Element: FolderProduction

Table 8-222: FolderProduction Element

Name	Data Type	Description
ModuleID ?	NMTOKEN	Identifies a particular folder module to be used. @ModuleID SHALL match Device/Module/@ModuleID .
ProductionType ?	enumeration	Indicates whether the product is collected or not. Allowed values are: Collect NonCollect

8.112 WindingParams

The parameters for the **Winding** process

Resource Properties

Resource referenced by:
Input of Processes: **Winding**
Output of Processes: —

Table 8-223: WindingParams Resource (Sheet 1 of 2)

Name	Data Type	Description
Copies ?	integer	Number of copies in one column that SHOULD be placed on a finished roll. At most one of @Copies , @Diameter or @Length SHOULD be specified.
Diameter ?	double	Outer diameter in points of the finished roll. At most one of @Copies , @Diameter or @Length SHOULD be specified.

Table 8-223: WindingParams Resource (Sheet 2 of 2)

Name	Data Type	Description
<i>Fixation</i> ?	NMTOKEN	<p>Method specifying how the Component is attached to the core.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>DoubleSidedTape</i> – Tape with adhesive on both sides. <i>Glue</i> <i>Label</i> – One of the output Component Resources (self-Adhesive labels) is used. <i>None</i> – No fixation is used. <i>SingleSidedTape</i> – Tape with adhesive on one side .
<i>Length</i> ?	double	Length in points of the Component to be placed on a finished roll. At most one of <i>@Copies</i> , <i>@Diameter</i> or <i>@Length</i> SHOULD be specified.

8.113 WrappingParams

WrappingParams defines the details of **Wrapping**. Details of the material used for **Wrapping** can be found either in the **MiscConsumable** or **Component** (Wrapper) Resources that are also an input of the **Wrapping** Process.

Resource Properties

Resource referenced by:

Intent Pairing:

Input of Processes:

Wrapping

Output of Processes:

—

Table 8-224: WrappingParams Resource

Name	Data Type	Description
<i>WrappingKind</i> ?	enumeration	<p><i>@WrappingKind</i> specifies the wrapping method.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Band</i> – The components are wrapped with a band. The material of the band is typically paper, plastic or rubber. <i>LooseWrap</i> – The wrap is loose around the component. <i>ShrinkWrap</i> – The wrap is shrunk around the component.
<i>WrappingKindDetails</i> ?	string	<p><i>@WrappingKindDetails</i> specifies additional details of the wrapping method.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>PaperBand</i> <i>PlasticBand</i> <i>RubberBand</i> <i>ShrinkWrap</i> <i>PaperWrap</i> <p>Note: site specific identifiers like "NeutralPaperBand" MAY also be provided.</p>

```

<Device Class="Implementation" ID="Link0003" Status="Available">
    <DeviceCap>
        <DisplayGroupPool>
            <DisplayGroup rRefs="btd cmp mag colorspace outputres">
                <Loc HelpText="Parameters for scanning configuration" Lang="en"
                     Value="ScanningParameters"/>
            </DisplayGroup>
        </DisplayGroupPool>
    </DeviceCap>
</Device>
<Device Class="Implementation" ID="Link0003" Status="Available">
    <DeviceCap>
        <FeaturePool>
            <EnumerationState
                AllowedValueList="Mono ColorTransparency Photo" ID="sm"
                HasDefault="false" MacroRefs="ScanModeMac" Name="ScanMode"
                UserDisplay="Display"/>
        </FeaturePool>
    </DeviceCap>
</Device>

<TestPool>
    <Test DescriptiveName="Transparencies cannot print duplex" Severity="Error">
        <and>
            <StringState AllowedRegExp="(Transparency)"
                         XPath="/XJDF/ResourceSet/Resource/Media/@MediaType"/>
            <StringState AllowedRegExp="(TwoSided)"
                         XPath="/XJDF/ParameterSet/Parameter/DigitalPrintingParams/@Sides"/>
        </and>
    </Test>
</TestPool>

<Device Class="Implementation" ID="Link0003" Status="Available">
    <DeviceCap>
        <ActionPool>
            <Action ID="MyAction" TestRef="ctcmp">
                <Loc HelpText="Only select CCITTFaxEncoding for 1 bit documents"
                     Lang="en" ShortValue="Ouch!"
                     Value="CCITTFaxEncoding not supported on
                            grayscale images"/>

```

```

        </Action>
    </ActionPool>
    <TestPool>
        <Test ID="ctcmp">
            <!-- Can't CCITT compress anything but 1 bit grayscale -->
            <and>
                <not>
                    <TestRef rRef="is1bit"/>
                </not>
                <EnumerationEvaluation ValueList="CCITTFaxEncode" rRef="cmp"/>
            </and>
        </Test>
        <Test ID="is1bit">
            <IntegerEvaluation ValueList="1" rRef="btd"/>
        </Test>
    </TestPool>
</DeviceCap>
</Device>

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.4" Version="1.4"
     TimeStamp="2005-04-05T16:45:43+02:00" SenderID="Controller"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<Query ID="DeviceQuery" Type="KnownDevices" xsi:type="QueryKnownDevices">
    <DeviceFilter DeviceDetails="Capability" Localization="fre"/>
</Query>
</JMF>
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="Scanner"
     TimeStamp="2005-06-05T16:45:43+02:00" MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<Response ID="xyz" Type="KnownDevices" refID="DeviceQuery"
           xsi:type="ResponseKnownDevices" >
    <DeviceList>
        <DeviceInfo DeviceStatus="Idle">
            <Device Class="Implementation" DeviceID="Joe the Drum"
                   KnownLocalizations="En Fre" ModelName="Bongo" >
                <DeviceCap GenericAttributes="ID Class SettingsPolicy
                                         BestEffortExceptions OperatorInterventionExceptions
                                         MustHonorExceptions PartIDKeys DocIndex"
                           Lang="Fre" Type="Scanning">
                    <!-- the scanner takes a minute to set up and scans an average
                        of 2 sheets a min. -->
                    <Performance AverageAmount="120" AverageSetup="PT2M"
                                  Name="ExposedMedia"/>
                <DevCaps Name="NodeInfo">
                    <DevCap>
                        <!--NodeInfo only supports JobPriority and
                            TargetRoute Attributes -->
                        <StringState Name="TargetRoute" HasDefault="false"/>
                        <IntegerState Name="JobPriority" HasDefault="false"/>
                    </DevCap>
                </DevCaps>
                <DevCaps Name="ExposedMedia">
                    <DevCap>
                        <!-- ExposedMedia restrictions -->
                        <DevCap Name="Media">
                            <NameState DefaultValue="Sheet" Name="MediaUnit"/>

```

```

<XYPairState AllowedValueMax="600 1200"
              AllowedValueMin="0 0"
              Name="Dimension" HasDefault="false"/>
</DevCap>
</DevCap>
</DevCaps>
<DevCaps Name="ScanParams">
    <Loc HelpText="Les parametres pour commander le
                  procede de balayage."
         Value="Les parametres de module de balayage"/>
<DevCap>
    <!-- Black and white 1 bit mode -->
    <IntegerState AllowedValueMax="1" AllowedValueMin="1"
                  DefaultValue="8" Name="BitDepth"/>
    <EnumerationState AllowedValueList="CCITTFaxEncode None"
                      Name="CompressionFilter" HasDefault="false">
        <Loc HelpText="Choisissez la compression pour reduire la
                      taille de donnees."
            Value="La compression de donnees"/>
        <ValueLoc Value="CCITTFaxEncode">
            <Loc Value="Compression de CCITT Fax"/>
        </ValueLoc>
        <ValueLoc Value="None">
            <Loc Value="Aucun compression"/>
        </ValueLoc>
    </EnumerationState>
    <NumberState AllowedValueMax="10" AllowedValueMin="1.e-002"
                 Name="Magnification" HasDefault="false">
        <Loc ShortValue="Rapport optique"
             Value="Rapport de rapport optique d'image"/>
    </NumberState>
    <EnumerationState AllowedValueList="GrayScale"
                      Name="OutputColorSpace" HasDefault="false">
        <Loc ShortValue="Format de couleur"
             Value="Configurez le format de couleur de
                   module de balayage"/>
        <ValueLoc Value="GrayScale">
            <Loc Value="echelle de gris"/>
        </ValueLoc>
    </EnumerationState>
    <XYPairState DefaultValue="2400 2400"
                  Name="OutputResolution">
        <Loc ShortValue="resolution"
             Value="Resolution de module de balayage"/>
    </XYPairState>
</DevCap>
<DevCap>
    <!-- Grayscale 12 bit mode -->
    <IntegerState AllowedValueMax="12" AllowedValueMin="12"
                  DefaultValue="8" Name="BitDepth">
        <Loc Value="Le profondeur de bit"/>
    </IntegerState>
    <EnumerationState
                  AllowedValueList="FlateEncode DCTEncode None"
                  Name="CompressionFilter" HasDefault="false">
        <Loc HelpText="Choisissez la compression pour
                      reduire la taille de donnees."
            Value="La compression de donnees"/>

```

```

<ValueLoc Value="FlateEncode">
    <Loc Value="Compression de Flate"/>
</ValueLoc>
<ValueLoc Value="DCTEEncode">
    <Loc Value="Compression de DCTE"/>
</ValueLoc>
<ValueLoc Value="None">
    <Loc Value="Aucun compression"/>
</ValueLoc>
</EnumerationState>
<NumberState AllowedValueMax="10" AllowedValueMin="0.001"
             Name="Magnification" DefaultValue="1.0">
    <Loc ShortValue="Rapport optique"
          Value="Rapport de rapport optique d'image"/>
</NumberState>
<EnumerationState AllowedValueList="GrayScale"
                   Name="OutputColorSpace" HasDefault="false">
    <Loc ShortValue="Format de couleur"
          Value="Configurez le format de couleur de
          module de balayage"/>
    <ValueLoc Value="GrayScale">
        <Loc Value="Echelle de gris"/>
    </ValueLoc>
</EnumerationState>
<XYPairState AllowedValueMax="2400 2400"
             AllowedValueMin="100 100" DefaultValue="600 600"
             Name="OutputResolution">
    <Loc ShortValue="resolution"
          Value="Resolution de module de balayage"/>
</XYPairState>
</DevCap>
<DevCap>
    <!-- Color 10 bit mode -->
    <IntegerState AllowedValueMax="10" AllowedValueMin="10"
                  DefaultValue="8" Name="BitDepth">
        <Loc Value="Le profondeur de bit"/>
    </IntegerState>
    <EnumerationState
                  AllowedValueList="FlateEncode DCTEEncode None"
                  Name="CompressionFilter">
        <Loc HelpText="Choisissez la compression pour reduire
                      la taille de donnees."
            Value="La compression de donnees"/>
        <ValueLoc Value="FlateEncode">
            <Loc Value="Compression de Flate"/>
        </ValueLoc>
        <ValueLoc Value="DCTEEncode">
            <Loc Value="Compression de DCTE"/>
        </ValueLoc>
        <ValueLoc Value="None">
            <Loc Value="Aucun compression"/>
        </ValueLoc>
    </EnumerationState>
    <NumberState AllowedValueMax="10" AllowedValueMin="1.e-002"
                 Name="Magnification">
        <Loc ShortValue="Rapport optique"
              Value="Rapport de rapport optique d'image"/>
    </NumberState>

```

```

<EnumerationState AllowedValueList="CMYK RGB LAB"
                   Name="OutputColorSpace">
  <Loc ShortValue="Format de couleur"
        Value="Configurez le format de couleur de
        module de balayage"/>
  <ValueLoc Value="CMYK">
    <Loc Value="Couleur de CMYK"/>
  </ValueLoc>
  <ValueLoc Value="RGB">
    <Loc Value="Couleur de RGB"/>
  </ValueLoc>
  <ValueLoc Value="LAB">
    <Loc Value="Couleur de LAB"/>
  </ValueLoc>
</EnumerationState>
<XYPairState AllowedValueMax="2400 2400"
              AllowedValueMin="100 100"
              DefaultValue="600 600" Name="OutputResolution">
  <Loc ShortValue="resolution"
        Value="Resolution de module de balayage"/>
</XYPairState>
</DevCap>
</DevCaps>
</DeviceCap>
</Device>
</DeviceInfo>
</DeviceList>
</Response>
</JMF>

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="Controller"
     TimeStamp="2005-04-05T16:45:43+02:00" MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
  <Query ID="DeviceQuery" Type="KnownDevices" xsi:type="QueryKnownDevices">
    <DeviceFilter DeviceDetails="Capability" Localization="en"/>
  </Query>
</JMF>

<JMF SenderID="Scanner"TimeStamp="2004-10-17T14:30:47Z"
      xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.4" Version="1.4"
      DescriptiveName="Example from JDF 1.2 Spec Document"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<Response ID="xyz" Type="KnownDevices" refID="DeviceQuery" ReturnCode="0"
           Acknowledged="false" xsi:type="ResponseKnownDevices" >
  <DeviceList>
    <DeviceInfo DeviceStatus="Idle">
      <Device DeviceID="Joe the Drum" ModelName="Bongo">
        <DeviceCap GenericAttributes="ID Class SettingsPolicy
          BestEffortExceptions OperatorInterventionExceptions
          MustHonorExceptions PartIDKeys DocIndex"
          Type="Scanning" CombinedMethod="None"
          ExecutionPolicy="AllFound">
          <Performance AverageAmount="120.0" Name="ExposedMedia" />
        <FeaturePool>
          <EnumerationState MinOccurs="1"
            AllowedValueList="Mono ColorTransparency Photo"
            UserDisplay="Display" Editable="true" ID="sm"
            ListType="SingleValue" HasDefault="true" Name="ScanMode">

```

```

        DevNS="http://www.CIP4.org/JDFSchema_1_1" MaxOccurs="1"
        MacroRefs="ScanModeMacro" />
    </FeaturePool>
    <DisplayGroupPool>
        <DisplayGroup rRefs="btd cmp mag colorspace outputres">
            <Loc HelpText="Parameters for scanning configuration"
                Lang="en" ShortValue="ScanningParameters" />
        </DisplayGroup>
    </DisplayGroupPool>
    <ActionPool>
        <Action Severity="Error" TestRef="BD-bw" ID="BD-bw-action">
            <Loc HelpText="For 1 bit grayscale, please select
                CCITTFaxEncoding"
                Lang="en" ShortValue="Ouch!"
                Value="Flate and DCT Encoding not allowed
                    on 1 bit images" />
        </Action>
        <Action Severity="Error" TestRef="ctcmp" ID="ctcmp-action">
            <Loc HelpText="Only select CCITTFaxEncoding for
                1 bit documents"
                Lang="en" ShortValue="Ouch!"
                Value="CCITTFaxEncoding not supported on
                    grayscale images" />
        </Action>
        <Action Severity="Error" TestRef="cd" ID="cd-action">
            <Loc HelpText="Choose a bit depth of 10 or less
                for color images"
                Lang="en" ShortValue="Ouch!"
                Value="Bit depths higher than 10 are not
                    supported for color" />
        </Action>
    </ActionPool>
    <TestPool>
        <Test ID="iscolor">
            <EnumerationEvaluation
                ValueList="RGB LAB CMYK" rRef="colorspace" />
        </Test>
        <Test ID="islbit">
            <IntegerEvaluation ValueList="1" rRef="btd" />
        </Test>
        <Test ID="BD-bw">
            <and>
                <TestRef rRef="islbit" />
                <EnumerationEvaluation
                    ValueList="FlateEncode DCTEncode"
                    rRef="cmp" />
            </and>
        </Test>
        <Test ID="ctcmp">
            <and>
                <not>
                    <TestRef rRef="islbit" />
                </not>
                <EnumerationEvaluation ValueList="CCITTFaxEncode"
                    rRef="cmp" />
            </and>
        </Test>
        <Test ID="cd">

```

```

<and>
    <TestRef rRef="iscolor" />
    <IntegerEvaluation ValueList="1 10" rRef="btd" />
</and>
</Test>
</TestPool>
<MacroPool>
    <macro ID="ScanModeMacro">
        <choice>
            <when>
                <EnumerationEvaluation ValueList="Mono" rRef="sm" />
                <set rRef="btd">
                    <FeatureAttribute CurrentValue="1" />
                </set>
                <set rRef="colorspace">
                    <FeatureAttribute CurrentValue="GrayScale" />
                </set>
                <set rRef="outputres">
                    <FeatureAttribute CurrentValue="1200 1200" />
                </set>
            </when>
            <when>
                <EnumerationEvaluation ValueList="ColorTransparency"
                    rRef="sm" />
                <set rRef="btd">
                    <FeatureAttribute CurrentValue="8" />
                </set>
                <set rRef="colorspace">
                    <FeatureAttribute CurrentValue="RGB" />
                </set>
                <set rRef="outputres">
                    <FeatureAttribute CurrentValue="600 600" />
                </set>
            </when>
            <when>
                <EnumerationEvaluation ValueList="Photo" rRef="sm" />
                <set rRef="btd">
                    <FeatureAttribute CurrentValue="10" />
                </set>
                <set rRef="colorspace">
                    <FeatureAttribute CurrentValue="LAB" />
                </set>
                <set rRef="outputres">
                    <FeatureAttribute CurrentValue="200 200" />
                </set>
            </when>
        </choice>
    </macro>
</MacroPool>
<DevCaps Required="false" Context="Resource"
    DevNS="http://www.CIP4.org/JDFSchema_1_1"
    Availability="Installed"
    Name="NodeInfo" ResourceUpdate="None">
    <DevCap MinOccurs="1" Name="NodeInfo"
        DevNS="http://www.CIP4.org/JDFSchema_1_1" MaxOccurs="1">
        <StringState UserDisplay="Display"
            DevNS="http://www.CIP4.org/JDFSchema_1_1"
            Editable="true" MinOccurs="1" MaxOccurs="1">

```

```

        Name="TargetRoute" HasDefault="true"
        ListType="SingleValue" />
<IntegerState Name="JobPriority"
    DevNS="http://www.CIP4.org/JDFSchema_1_1"
    Editable="true" MinOccurs="1" MaxOccurs="1"
    UserDisplay="Display" HasDefault="true"
    ListType="SingleValue" />
</DevCap>
</DevCaps>
<DevCaps Required="false" ResourceUpdate="None" Context="Resource"
    Availability="Installed" Name="ExposedMedia"
    DevNS="http://www.CIP4.org/JDFSchema_1_1">
<DevCap MinOccurs="1" Name="ExposedMedia"
    DevNS="http://www.CIP4.org/JDFSchema_1_1" MaxOccurs="1">
<DevCap MinOccurs="1" Name="Media"
    DevNS="http://www.CIP4.org/JDFSchema_1_1"
    MaxOccurs="1">
<NameState MinOccurs="1" DefaultValue="Sheet"
    UserDisplay="Display" Editable="true"
    ListType="SingleValue" HasDefault="true"
    Name="MediaUnit" MaxOccurs="1"
    DevNS="http://www.CIP4.org/JDFSchema_1_1"/>
<XYPairState MinOccurs="1" UserDisplay="Display"
    Editable="true" AllowedValueMax="600.0 1200.0"
    ListType="SingleValue" HasDefault="true"
    Name="Dimension" AllowedValueMin="0.0 0.0"
    DevNS="http://www.CIP4.org/JDFSchema_1_1"
    MaxOccurs="1" />
</DevCap>
</DevCap>
</DevCaps>
<DevCaps Required="false" Context="Resource"
    DevNS="http://www.CIP4.org/JDFSchema_1_1"
    Availability="Installed" Name="ScanParams"
    ResourceUpdate="None">
<DevCap MinOccurs="1" Name="ScanParams"
    DevNS="http://www.CIP4.org/JDFSchema_1_1" MaxOccurs="1">
<IntegerState MinOccurs="1" DefaultValue="1"
    AllowedValueList="1 4 8 10 12" UserDisplay="Hide"
    ActionRefs="BD-bw ctcmp cd" Editable="true"
    ID="btd" ListType="SingleValue" HasDefault="true"
    Name="BitDepth" MaxOccurs="1"
    DevNS="http://www.CIP4.org/JDFSchema_1_1"/>
<EnumerationState ActionRefs="BD-bw ctcmp" MinOccurs="1"
    AllowedValueList=
        "CCITTFaxEncode FlateEncode DCTEncode None"
    UserDisplay="Hide" Editable="true" ID="cmp"
    ListType="SingleValue" HasDefault="true"
    Name="CompressionFilter" MaxOccurs="1"
    DevNS="http://www.CIP4.org/JDFSchema_1_1" />
<NumberState MinOccurs="1" UserDisplay="Display"
    Editable="true" ID="mag" ListType="SingleValue"
    HasDefault="true" AllowedValueMax="100.0"
    AllowedValueMin="0.01" MaxOccurs="1"
    Name="Magnification"
    DevNS="http://www.CIP4.org/JDFSchema_1_1"/>
<EnumerationState ActionRefs="cd" MinOccurs="1"
    AllowedValueList="GrayScale CMYK RGB LAB"

```

```

        UserDisplay="Display" Editable"true"
        ID="colorspace" ListType"SingleValue"
        HasDefault"true" Name"OutputColorSpace"
        MaxOccurs"1"
        DevNS="http://www.CIP4.org/JDFSchema_1_1" />
    <XYPairState MinOccurs"1" DefaultValue"600.0 600.0"
        AllowedValueList"100.0 100.0 300.0 300.0 600.0 600.0
            1200.0 1200.0 2400.0 2400.0"
        UserDisplay="Display" Editable"true" ID"outputs"
        ListType"SingleValue" HasDefault"true"
        Name"OutputResolution" MaxOccurs"1"
        DevNS="http://www.CIP4.org/JDFSchema_1_1" />
    </DevCap>
</DevCaps>
</DeviceCap>
</Device>
</DeviceInfo>
</DeviceList>
</Response>
</JMF>

<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID"GoodScan"
    Status"Waiting" Type"Scanning" JobPartID"ID300" Version"1.4">
<ResourcePool>
    <ScanParams BitDepth"8" Class"Parameter" ID"Link0007"
        OutputColorSpace"RGB" OutputResolution"600. 600." Status"Available"/>
    <ExposedMedia Class"Handling" ID"Link0008" Status"Available">
        <Media Dimension"425.196850394 566.929133858"/>
    </ExposedMedia>
    <RunList Class"Parameter" ID"Link0014" Status"Available"/>
</ResourcePool>
<ResourceLinkPool>
    <ScanParamsLink Usage"Input" rRef"Link0007"/>
    <ExposedMediaLink Usage"Input" rRef"Link0008"/>
    <RunListLink Usage"Output" rRef"Link0014"/>
</ResourceLinkPool>
</JDF>

<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID"BadScan" Status"Waiting"
    Type"Scanning" JobPartID"ID300" Version"1.4">
<ResourcePool>
    <ScanParams BitDepth"8" Class"Parameter" ID"Link0012"
        Magnification"1000. 1000."
        OutputColorSpace"RGB" OutputResolution"600. 600." Status"Available"/>
    <ExposedMedia Class"Handling" ID"Link0013" Status"Available">
        <Media Dimension"425.196850394 566.929133858"/>
    </ExposedMedia>
    <RunList Class"Parameter" ID"Link0014" Status"Available"/>
</ResourcePool>
<ResourceLinkPool>
    <ScanParamsLink Usage"Input" rRef"Link0012"/>
    <ExposedMediaLink Usage"Input" rRef"Link0013"/>
    <RunListLink Usage"Output" rRef"Link0014"/>
</ResourceLinkPool>
</JDF>

<Device Class"Implementation" ID"Link0003" Status"Available">
    <DeviceCap>
        <TestPool>

```

```

<Test ID="PT01">
  <IsPresentEvaluation>
    <BasicPreflightTest Name="TrappedKey"/>
  </IsPresentEvaluation>
</Test>
</TestPool>
</DeviceCap>
</Device>
<Device Class="Implementation" ID="Link0003" Status="Available">
  <DeviceCap>
    <TestPool>
      <Test ID="PT02">
        <EnumerationEvaluation ValueList="Unknown">
          <BasicPreflightTest Name="TrappedKey"/>
        </EnumerationEvaluation>
      </Test>
    </TestPool>
  </DeviceCap>
</Device>
<Device Class="Implementation" ID="Link0003" Status="Available">
  <DeviceCap>
    <TestPool>
      <Test ID="PT01">
        <RectangleEvaluation ValueList="0 0 10 10">
          <BasicPreflightTest Name="BoxToBoxDifference">
            <PreflightArgument>
              <BoxToBoxDifference FromBox="TrimBox"
                ToBox="BleedBox"/>
            </PreflightArgument>
          </BasicPreflightTest>
        </RectangleEvaluation>
      </Test>
    </TestPool>
  </DeviceCap>
</Device>

```

Chapter 10 Subelements

The elements in this chapter are subelements that can occur in multiple Resources. They are not Resources and are therefore never directly linked to Processes.

10.1 Address

Definition of an address. The structure is derived from the vCard format and, therefore, is comprised of all address subtypes (ADR:) of the delivery address of the vCard format. The corresponding XML types of the vCard are quoted in the table.

Element Properties

Element referenced by: Location, Contact, Person

Table 10-1: Address Element

Name	Data Type	Description
<i>City</i> ?	string	City or locality of address (vCard: ADR:locality).
<i>Country</i> ?	string	Country of address (vCard: ADR:country).
<i>CountryCode</i> ?	string	Country of address. Values include those from: [ISO3166-1:1997] Note: countries are represented as two-character codes
<i>PostalCode</i> ?	string	Zip code or postal code of address (vCard: ADR:pcode).
<i>PostBox</i> ?	string	Post office address (vCard: ADR:pobox. For example: P.O. Box 101).
<i>Region</i> ?	string	State or province (vCard: ADR:region).
<i>Street</i> ?	string	Street address (vCard: ADR:street).
<i>ExtendedAddress</i> ?	text element	Extended address (vCard: ADR:extadd. For example: Suite 245).

10.2 ApprovalPerson

Table 10-2: ApprovalPerson Element

Name	Data Type	Description
<i>ApprovalRole</i> ?	enumeration	Role of the ApprovalPerson. Allowed values are: <i>Approvator</i> – The decision of this approver immediately overrides the decisions of the other approvers and ends the approval cycle. The "Approvator" NEED NOT sign for the approval to become valid. <i>Informative</i> – The approver is informed of the Approval Process, but the approval is still valid, even without his approval. <i>Obligated</i> – The approver SHALL sign the approval.
<i>ApprovalRoleDetails</i> ?	string	Additional details on the @ApprovalRole.
<i>ContactRef</i>	IDREF	Contact (e.g., a customer, printer or manager) who SHALL sign the approval. There SHALL be a Contact [contains (@ContactTypes, "Approver")].

10.3 AutomatedOverPrintParams

This provides controls for the automated selection of overprinting of black text or graphics. *@RGBGray2Black* and *@RGBGray2BlackThreshold* in *ColorSpaceConversion*/ are used by the *ColorSpaceConversion* Process in determining the allocation of RGB values to the black (K) channel. After the *ColorSpaceConversion* Process is completed, then the *Rendering* or *Separation* Process uses **AutomatedOverPrintParams** to determine overprint behavior for the previously determined black (K) channel.

Element Properties

Element referenced by: **RenderingParams, SeparationControlParams**

Table 10-3: AutomatedOverPrintParams Element

Name	Data Type	Description
<i>KnockOutCMYKWhite</i> ?	boolean	Graphic objects defined in DeviceCMYK, where all colorant values are <0.001 SHALL be knocked out, even when set to overprint and when the PDF overprint mode is set to 1.
<i>OverPrintBlackLineArt</i> ?	boolean	Indicates whether overprint is to be set to "true" for black line art (i.e., vector elements other than text). If "true", overprint of black line art is applied regardless of any values in the PDL. If "false", <i>@LineArtBlackLevel</i> is ignored and PDL line art overprint operators are processed.
<i>LineArtBlackLevel</i> ?	double	A value between 0.0 and 1.0 which indicates the minimum black level for the stroke or fill colors that cause the line art to be set to overprint. Defaults to the value of <i>@TextBlackLevel</i> .
<i>OverPrintBlackText</i> ?	boolean	Indicates whether overprint is to be set to "true" for black text. If "true", overprint of black text is applied regardless of any values in the PDL. If "false", <i>@TextSizeThreshold</i> and <i>@TextBlackLevel</i> are ignored and PDL text overprint operators are processed.
<i>TextBlackLevel</i> ?	double	A value between 0.0 and 1.0 which indicates the minimum black level for the text stroke or fill colors that cause the text to be set to overprint.
<i>TextSizeThreshold</i> ?	integer	Indicates the point size for text below which black text will be set to overprint. For asymmetrically scaled text, the minimum point size between both axes will be used. If not specified, all text is set to overprint.

10.4 BarcodeCompParams

BarcodeCompParams specifies the technical compensation parameters for barcodes.

Element Properties

Element referenced by: **BarcodeReproParams**

Table 10-4: BarcodeCompParams Element (Sheet 1 of 2)

Name	Data Type	Description
<i>CompensationProcess</i>	enumeration	Process that is bar width spread is compensated for. Allowed values are: <i>Printing</i> <i>Platemaking</i>

Table 10-4: BarcodeCompParams Element (Sheet 2 of 2)

Name	Data Type	Description
<i>CompensationValue</i> ?	double	The width of the bars is reduced by this amount in micron to compensate for technical spread.

10.5 BarcodeReproParams

BarcodeReproParams specifies the reproduction parameters for barcodes.

Element Properties

Element referenced by: /Content/BarcodeProductionParams, , DeviceMark

Table 10-5: BarcodeReproParams Element

Name	Data Type	Description
<i>BearerBars</i> ?	enumeration	Indicates the policy how to generate bearer bars. (ITF). Allowed values are: <i>None</i> <i>TopBottom</i> <i>Box</i> <i>BoxHMarks</i>
<i>Height</i> ?	double	The height of the bars of a linear barcode.
<i>Magnification</i> ?	double	The magnification factor for linear barcodes.
<i>Masking</i> ?	enumeration	Indicates the properties of the mask around the graphical content of the barcode that masks out all underlying graphics. Allowed values are: <i>None</i> – No masking, barcode is put on top of underlying graphics. <i>WhiteBox</i> – An area of the underlying graphics is masked out (the white box) and the barcode is put on top of this masked area. The area of the white box is the box enclosing all artwork of the barcode, excluding optional human readable text. This would enclose bearer bars, quiet zones and non-optional human readable text (UPC and EAN barcodes).
<i>ModuleHeight</i> ?	double	The Y size in micron of an Element of a 2D barcode (e.g., PDF417). For DATAMATRIX, Y Dimension MAY be omitted (X Dimension = Y Dimension).
<i>ModuleWidth</i> ?	double	The X size in micron of an Element of a 2D barcode such as DATAMATRIX or PDF417.
<i>Ratio</i> ?	double	the ratio between the width of the narrow bars and the wide bars for those barcodes where ratio the width of the wide bars and narrow bars MAY vary.
BarcodeCompParams *	element	Parameters for bar width compensation. The total reduction of bar width is the sum of all BarcodeCompParams/@CompensationValue.

10.6 ColorantAlias

ColorantAlias is an Element that specifies a replacement colorant name string to be used instead of one or more named colorant strings. **Element Properties**

Element referenced by: ColorantControl

Table 10-6: ColorantAlias Element

Name	Data Type	Description
<i>ColorantName</i>	string	The name of the colorant to be replaced in PDL files.
<i>RawName</i> ?	hexBinary	<i>@RawName</i> represents the original 8-bit byte stream of the color specified in <i>@ColorantName</i> . Used to transport the original byte representation of a color name when moving XJDF tickets between computers with different locales..
<i>ReplacementColorantName</i>	NMTOKEN	The value of Color ../ <i>@Separation</i> to be substituted for the colorant name string <i>@ColorantName</i> .

Example 10-1: ColorantAlias/@RawName

TBD 2.x Example.

```
<ColorantAlias Class="Parameter" ID="r000004" RawNames="4772FC6E 6772FC6E"
    ReplacementColorantName="Green" Status="Available">
    <!-- ColorantAlias that maps the additional representation (grün, Grün)
        to the predefined separation Green -->
    <SeparationSpec Name="Grün"/>
    <SeparationSpec Name="grün"/>
</ColorantAlias>
```

10.7 ColorCorrectionOp

Element PropertiesElement referenced by: **ColorCorrectionParams****Table 10-7: ColorCorrectionOp Element (Sheet 1 of 2)**

Name	Data Type	Description
<i>AdjustContrast</i> ?	double	Specifies the L*a*b* contrast adjustment in the range -100 (minimum contrast for the system (i.e., a solid midtone gray color)) to + 100 (maximum contrast for the system (i.e., either use full color (the maximum is restricted by the system ink limit) or no color for each of Cyan, Magenta, Yellow and Black)). Increasing the contrast value increases the variation between light and dark areas and decreasing the contrast value decreases the variation between light and dark areas. See explanation above.
<i>AdjustCyanRed</i> ?	double	Specifies the L*a*b* adjustment in the Cyan/Red axis in the range -100 (maximum Cyan cast for the system) to + 100 (maximum Red cast for the system) while maintaining lightness. See explanation above.
<i>AdjustHue</i> ?	double	Specifies the change in the L*a*b* hue in the range -180 to +180 of all colors by the specified number of degrees of the color circle. See explanation above.
<i>AdjustLightness</i> ?	double	Specifies the decrease or increase of the L*a*b* lightness in the range -100 (minimum lightness for the system (i.e., black)) to + 100 (maximum lightness for the system (i.e., white)). Increasing the lightness value causes the output to appear lighter and decreasing the lightness value causes the output to appear darker. See explanation above.

Table 10-7: ColorCorrectionOp Element (Sheet 2 of 2)

Name	Data Type	Description
<i>AdjustMagentaGreen</i> ?	double	Specifies the L*a*b* adjustment in the Magenta/Green axis in the range -100 (maximum Magenta cast for the system) to + 100 (maximum Green cast for the system) while maintaining lightness. See explanation above.
<i>AdjustSaturation</i> ?	double	Specifies the increase or decrease of the L*a*b* color saturation in the range -100 (minimum saturation for the system) to +100 (maximum saturation for the system). Increasing the saturation value causes the output to contain more vibrant colors and decreasing the saturation value causes the output to contain more pastel and gray colors. See explanation above.
<i>AdjustYellowBlue</i> ?	double	Specifies the L*a*b* adjustment in the Yellow/Blue axis in the range -100 (maximum Yellow cast for the system) to + 100 (maximum Blue cast for the system) while maintaining lightness. See explanation above.
<i>ObjectTags</i> ?	NMTOKENS	<p>Tags associated with individual objects that this <i>ColorCorrectionOp</i> SHALL be applied to. Each tag specified in <i>@ObjectTags</i> is logically anded with the object type(s) specified by <i>@SourceObjects</i>, enabling first qualification by object type (such as image), and then tags associated with those objects.</p> <p>The values of <i>@ObjectTags</i> depends on the PDL that the color correction is applied to.</p> <p><i>@ObjectTags</i> SHALL apply only to objects whose tag pool includes all the tags in the value of <i>@ObjectTags</i>.</p>
<i>SourceObjects</i> ?	enumerations	<p>Identifies which class(es) of incoming graphical objects will be operated on.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>All</i> <i>ImagePhotographic</i> – Contone images. <i>ImageScreenShot</i> – Images largely comprised of rasterized vector art. <i>LineArt</i> – Vector objects other than text. <i>SmoothShades</i> – Gradients and blends. <i>Text</i>
<i>FileSpec (AbstractProfile)</i> ?	element	A FileSpec Resource pointing to an abstract ICC profile that has been devised to apply a preference adjustment. See explanation of adjustment at the beginning of this section.
<i>FileSpec (DeviceLinkProfile)</i> ?	element	A FileSpec Resource pointing to an ICC profile that describes the characterization of an abstract profile for specifying a preference adjustment. See explanation of adjustment at the beginning of this section.

10.8 ColorSpaceConversionOp

The *ColorSpaceConversionOp* Element identifies a type of object, defines the source color space for that type of object, and specifies the behavior of the conversion operation for that type of object. Many of these Attribute descriptions refer to ICC Color Profiles[ICC.1]. See also the International Color Consortium (ICC) Web site at <http://www.color.org>.

Element PropertiesElement referenced by: **ColorSpaceConversionParams****Table 10-8: ColorSpaceConversionOp Element (Sheet 1 of 4)**

Name	Data Type	Description
<i>ObjectTags</i> ?	NMTOKENS	<p>Tags associated with individual objects that this ColorSpaceConversionOp SHALL be applied to. Each tag specified in <i>@ObjectTags</i> is logically anded with the object type(s) specified by <i>@SourceObjects</i>, enabling first qualification by object type (such as image), and then tags associated with those objects.</p> <p>The values of <i>@ObjectTags</i> depends on the PDL that the color space conversion is applied to.</p> <p><i>@ObjectTags</i> SHALL apply only to objects whose tag pool includes all the tags in the value of <i>@ObjectTags</i>.</p>
<i>Operation</i> ?	enumeration	<p>Controls which of five functions the color space conversion utility performs.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Convert</i> – Transforms graphical elements to final target color space. <i>Tag</i> – Associates appropriate working space profile with uncharacterized graphical element. <i>Untag</i> – Removes all profiles and color characterizations from graphical elements. <i>Retag</i> – Equivalent to a sequence of "<i>Untag</i>" → "<i>Tag</i>", where the "<i>Untag</i>" → "<i>Tag</i>" sequence is only applied to those objects selected by this ColorSpaceConversionOp. <i>ConvertIgnore</i> – Equivalent to a sequence of "<i>Untag</i>" → "<i>Convert</i>".
<i>PreserveBlack</i> ?	boolean	Controls how the tints of black (K in CMYK) are to be handled. If <i>@PreserveBlack</i> is "false", these colors are processed through the standard ICC workflow. If <i>@PreserveBlack</i> is "true", these colors are to be converted into other shades of black. The algorithm is implementation-specific.
<i>RenderingIntent</i> ?	enumeration	<p>Identifies the rendering intent to be applied when rendering the objects selected by this ColorSpaceConversionOp.</p> <p>Allowed values are (ICC-defined [ICC.1] rendering intent values):</p> <ul style="list-style-type: none"> <i>Saturation</i> <i>Perceptual</i> <i>RelativeColorimetric</i> <i>AbsoluteColorimetric</i> <i>ColorSpaceDependent</i> – The rendering intent is dependent on the color space. The dependencies are implementation specific.

Table 10-8: ColorSpaceConversionOp Element (Sheet 2 of 4)

Name	Data Type	Description
<i>RGBGray2Black</i> ?	boolean	This feature controls what happens to gray values (R = G = B) when converting from RGB to CMYK or the incoming graphical objects indicated by <i>@SourceObjects</i> . In the case of MS Office applications and screen dumps, there are a number of gray values in the images and line art. Printers do not want to have CMY under the K because it creates registration problems. They prefer to have K only, so the printer converts the gray values to K. Gray values that exceed the <i>@RGBGray2BlackThreshold</i> are not converted. <i>@RGBGray2Black</i> and <i>@RGBGray2BlackThreshold</i> are used by the ColorSpaceConversion Process in determining how to allocate RGB values to the black (K) channel. After the ColorSpaceConversion Process is completed, the Rendering Process uses AutomatedOverPrintParams to determine overprint behavior for the previously determined black (K) channel.
<i>RGBGray2BlackThreshold</i> ?	double	A value between "0.0" and "1.0" which specifies the threshold value above which the Device SHALL NOT convert gray (R = G = B) to black (K only) when <i>@RGBGray2Black</i> is "true". So a "0" value means convert only R = G = B = 0 (black) to K only. A value of "1" specifies that all values of R = G = B are converted to K if <i>@RGBGray2Black</i> = "true".
<i>Séparations</i> ?	NMTOKENS	List of separations that specify on which separation(s) to operate when <i>@SourceCS</i> = "Separation".
<i>SourceCS</i>	enumeration	Identifies which of the incoming color spaces will be operated on. Allowed values are from: Table 10-10, "SourceCS Attribute Values". Note: see Table 10-11, "Mapping of SourceCS enumeration values to color spaces in the most common input file formats".
<i>SourceObjects</i> ?	enumerations	List of object Classes that identifies which incoming graphical objects will be operated on. Allowed values are: <i>All</i> <i>ImagePhotographic</i> – Contone images. <i>ImageScreenShot</i> – Images largely comprised of rasterized vector art. <i>LineArt</i> – Vector objects other than text. <i>SmoothShades</i> – Gradients and blends. <i>Text</i>

Table 10-8: ColorSpaceConversionOp Element (Sheet 3 of 4)

Name	Data Type	Description
<i>SourceRenderingIntent</i> ?	enumeration	<p>Identifies the rendering intent transform elements to be selected from the source profile that will be used to interpret objects of type identified by the <i>@SourceObjects</i> and <i>@SourceCS</i> Attributes.</p> <p>Default value is from: <i>@RenderingIntent</i>.</p> <p>Allowed values are (ICC-defined [ICC.1] rendering intent values):</p> <ul style="list-style-type: none"> <i>Saturation</i> <i>Perceptual</i> <i>RelativeColorimetric</i> <i>AbsoluteColorimetric</i> <p><i>ColorSpaceDependent</i> – The rendering intent is dependent on the color space. The dependencies are implementation specific.</p> <p>Note: The <i>@SourceRenderingIntent</i> will pertain to the source profile used in a particular ColorSpaceConversion Process (e.g., sources can be the native original color space, an intermediate working color space or an reference output simulation color space).</p>
<i>DeviceNSpace</i> ?	element	DeviceNSpace Resource that describe the DeviceN color space on which to operate when <i>@SourceCS</i> = "DeviceN". Individual colorant definitions for the colorant names given in DeviceNSpace are provided in the ColorantControl/ <i>@ColorRef</i> Attribute, which SHALL also be present
FileSpec <i>(AbstractProfile)</i> ?	element	A FileSpec Resource pointing to an ICC profile [ICC.1] that describes the characterization of an Abstract Profile for specifying a preference adjustment.
FileSpec <i>(DeviceLinkProfile)</i> *	element	<p>A FileSpec Resource pointing to an ICC profile file [ICC.1] that contains a Device Link transform.</p> <p>The Source colorspace of the referenced Device Link transform SHOULD match that of the profile identified by FileSpec (<i>PDLSourceProfile</i>) and the destination color space SHOULD match that of the destination profile identified by ColorSpaceConversionParams (if specified). Multiple Device Link ICC transforms should be provided where each transform specifies a different rendering intent. This is important in the case where multiple PDL content objects of the colorspace specify different rendering intents.</p> <p>Note: an ICC Device Link profile contains only one transform with one color rendering intent.</p> <p>Note: although the ICC specification refers to all ICC files as "profiles", a Device Link in actuality represents a single transform to be applied, and not a profile of a particular device colorspace. Thus these files are referred to as Device Link transforms in this specification.</p> <p>FileSpec</p>

Table 10-8: ColorSpaceConversionOp Element (Sheet 4 of 4)

Name	Data Type	Description
FileSpec (<i>PDLSourceProfile</i>)?	element	A FileSpec Resource describing an ICC profile that describes a profiled source space that this ColorSpaceConversionOp should operate on. When present, only objects that specify the <i>@SourceCS</i> along with the specified profile are selected. Note: The FileSpec/@UID Attribute can often be used to positively identify an ICC profile referenced in a PDL file when available (FileSpec/@UID corresponds to the ICC ProfileID field). In addition, FileSpec/@CheckSum may be used when only a checksum of the entire profile is available.
FileSpec (<i>SourceProfile</i>)?	element	A FileSpec Resource pointing to an ICC profile [ICC.1] that describes the assumed source color space. If <i>SourceProfileRef</i> is specified, it SHALL be used as the profile for the source object's color space during a "Convert", "Tag", or "Retag" operation, as specified by <i>@Operation</i> . <i>SourceProfileRef</i> SHALL be present for "Tag" or "Retag" operations, as specified by <i>@Operation</i> .
ScreenSelector ?	element	Links this ColorSpaceConversionOp to a given screening.

— Attribute: SourceCS**Table 10-9: SourceCS Attribute Values (Sheet 1 of 2)**

Value	Description
<i>All</i>	Operates on all source colorspaces. This is useful when specifying a Convert operation using all PDL source-supplied characterizations with an XJDF-supplied final target device profile.
<i>CalGray</i>	defines a calibrated Device independent representation of Gray.
<i>Calibrated</i>	Operates on " <i>CalGray</i> " and " <i>CalRGB</i> " color spaces.
<i>CalRGB</i>	defines a calibrated based Device independent representation of RGB.
<i>CIEBased</i>	Operates on CIE-based color spaces (" <i>CIEBasedA</i> ", " <i>CIEBasedABC</i> ", " <i>CIEBasedDEF</i> " and " <i>CIEBasedDEFG</i> ").
<i>CMYK</i>	Operates on all CMYK color spaces. This includes both characterized and uncharacterized CMYK color spaces.
<i>DeviceCMYK</i>	Operates on uncharacterized CMYK color spaces.
<i>DeviceGray</i>	Operates on uncharacterized Gray color spaces.
<i>DeviceN</i>	Identifies the source color encoding as a " <i>DeviceN</i> " color space. The specific " <i>DeviceN</i> " color space to operate on is defined in the <i>DeviceNSpace</i> Resource. If " <i>DeviceN</i> " is specified, then <i>DeviceNSpace</i> SHALL also be present.
<i>DeviceRGB</i>	Operates on uncharacterized RGB color spaces.
<i>DevIndep</i>	Operates on Device independent color spaces (equivalent to " <i>Calibrated</i> ", " <i>CIEBased</i> ", " <i>ICCBased</i> ", " <i>Lab</i> " or " <i>YUV</i> ").
<i>Gray</i>	Operates on all Gray color spaces. This includes both characterized and uncharacterized Gray color spaces.
<i>ICCBased</i>	Operates on color spaces defined using ICC profiles. The " <i>ICCBased</i> " value includes EPS, TIFF or PICT files with embedded ICC profiles. See [ICC.1]. Includes PDF Device color spaces that are characterized in Footnote 1 on page 1079 following Table 10-11.

Table 10-9: SourceCS Attribute Values (Sheet 2 of 2)

Value	Description
<i>ICCCMYK</i>	Operates on ICCBased color spaces with ICC CMYK profiles or DeviceCMYK having an ICC-based characterization. See Footnote 1 on page 1079 following Table 10-11.
<i>ICCGray</i>	Operates on ICCBased color spaces with ICC gray profiles or DeviceGray having an ICC-based characterization. See Footnote 1 on page 1079 following Table 10-11.
<i>ICCLAB</i>	Operates on ICC based Device independent representation of LAB
<i>ICCRGB</i>	Operates on ICCBased color spaces with ICC RGB profiles or DeviceRGB having an ICC-based characterization. See Footnote 1 on page 1079 following Table 10-11.
<i>Lab</i>	Operates on "Lab".
<i>RGB</i>	Operates on all RGB color spaces. This includes both characterized and uncharacterized RGB color spaces.
<i>Separation</i>	Operates on "Separation" color spaces (spot colors). The specific separation(s) to operate on are defined in the <i>@Separations</i> Attribute. If no <i>@Separations</i> is defined, the operation will operate on all the separation color spaces in the input RunList .
<i>YUV</i>	Operates on "YUV" (Also known as YCbCr). See [CCIR601-2]

Note:

The following table summarizes how the *@SourceCS* Attribute is mapped to/from different file formats.

Table 10-10: Mapping of SourceCS enumeration values to color spaces in the most common input file formats (Sheet 1 of 2)

SourceCS	File Format	Color Space
<i>Calibrated</i>	PDF (2)	CalGray, CalRGB
	PostScript (2)	n/a
	TIFF	n/a
<i>CIEBased</i>	PDF (2)	n/a
	PostScript (2)	CIEBasedABC, CIEBasedA, CIEBasedDEF and CIEBasedDEFG
	TIFF	n/a
<i>CMYK</i>	PDF (2)	DeviceCMYK (1) PDF ICCBased color spaces with ICC CMYK profiles. CIEBasedDEFG spaces that resolve to a characterized CMYK space.
	PostScript (2)	DeviceCMYK
	TIFF	PhotometricInterp = 5 Samples per pixel = 4
<i>DeviceCMYK</i>	PDF (2)	DeviceCMYK (1)
	PostScript (2)	DeviceCMYK
	TIFF	PhotometricInterp = 5 Samples per pixel = 4
<i>DeviceGray</i>	PDF (2)	DeviceGray (1)
	PostScript (2)	DeviceGray
	TIFF	PhotometricInterp = 0 or 1

Table 10-10: Mapping of SourceCS enumeration values to color spaces in the most common input file formats (Sheet 2 of 2)

SourceCS	File Format	Color Space
DeviceN	PDF (2)	DeviceN
	PostScript (2)	DeviceN
	TIFF	PhotometricInterp = 5, Samples per pixel = N
DeviceRGB	PDF (2)	DeviceRGB (1)
	PostScript	DeviceRGB
	TIFF	PhotometricInterp = 2
Gray	PDF (2)	DeviceGray (1) PDF ICCBased color spaces with ICC Gray profiles. CIEBasedA spaces that resolve to a characterized Gray space.
	PostScript (2)	DeviceGray
	TIFF	PhotometricInterp = 0 or 1
ICCBased	PDF (2)	ICCBased DeviceGray, DeviceCMYK, DeviceRGB (See Footnote 1 on page 1079)
	PostScript (2)	n/a
	PostScript/EPS	The EPS file has an embedded ICC profile.
	TIFF	The TIFF file has an embedded ICC profile.
	LAB	LAB
LAB	PostScript (2)	n/a
	TIFF	PhotometricInterp = 8 (CIELAB 1976 “normal” encoding) or PhotometricInterp = 9 (CIELAB 1976 using ICC profile v2 encoding).
	RGB	DeviceRGB (1) PDF ICCBased color spaces with ICC RGB profiles. CIEBasedDEF spaces that resolve to a characterized RGB space.
Separation	PostScript	DeviceRGB
	TIFF	PhotometricInterp = 2
	PDF (2)	Separation
YUV	PostScript (2)	Separation
	TIFF	PhotometricInterp = 5 (Applies only to one of the planes in the separated image.)
	PDF (2)	n/a
PostScript (2)	n/a	
	TIFF	PhotometricInterp = 6

- 1) In PDF, DeviceCMYK, DeviceRGB, and DeviceGray source colorspaces can be characterized through providing a DefaultCMYK, DefaultRGB, or DefaultGray resource specifying a profile to be associated with source objects in that color space. In which case, the resulting color space is considered characterized by **XJDF** operations.
- 2) Where a "Pattern" or "Indexed" color space has been used, the base color space is used to determine whether to apply this operation.

<ComChannel Class="Parameter" ID="cc000004" ChannelType="Phone"

```


    ChannelTypeDetails="Mobile" ChannelUsage="Business"
    Locator="tel:+44-07808-907-919" Status="Available"/>
<ComChannel Class="Parameter" ID="cc000004" ChannelType="InstantMessaging"
    ChannelTypeDetails="MyIMService" ChannelUsage="Private"
    Locator="123456789" Status="Available"/>


```

10.9 ConvertingConfig

The **ConvertingConfig** Element describes a range of Sheet sizes. that can be used for optimizing a **DieLayoutProduction** or a press sheet for **SheetOptimizing**.

Element Properties

Element referenced by: **DieLayoutProductionParams, SheetOptimizingParams**

Table 10-11: ConvertingConfig Element

Name	Data Type	Description
<i>MarginBottom</i> ?	double	The bottom margin for positioning the layout on the Sheet.
<i>MarginLeft</i> ?	double	The left margin for positioning the layout on the Sheet.
<i>MarginRight</i> ?	double	The right margin for positioning the layout on the Sheet.
<i>MarginTop</i> ?	double	The top margin for positioning the layout on the Sheet.
<i>SheetHeightMax</i> ?	double	The maximum Sheet height (pt)..
<i>SheetHeightMin</i> ?	double	The minimum Sheet height (pt).
<i>SheetWidthMax</i> ?	double	The maximum Sheet width (pt).
<i>SheetWidthMin</i> ?	double	The minimum Sheet width (pt).
Device *	element	The target devices (printing press, die cutter and further finishing equipment) corresponding to this configuration. Typically only the type of Device would be used (e.g., the model of the die cutter). If multiple Devices are specified, then the other Attributes in this Element SHALL apply to a production configuration that uses all specified Devices.
Media *	element	Reference to zero or more Media elements that are candidates for optimization. Note: this element allows a media database savvy consumer to loop over an explicit range of known materials rather than providing results based on a range of dimensions only.

10.10 Crease

Crease defines an individual crease line on a **Component**.

Element Properties

Element referenced by: **CreasingParams, FoldingParams**

Table 10-12: Crease Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Depth</i> ?	double	Depth of the crease, measured in microns [μm].
<i>StartPosition</i> ?	XYPair	Starting position of the tool.
<i>Travel</i> ?	double	Distance of the reference edge relative to <i>@From</i> .

Table 10-12: Crease Element (Sheet 2 of 2)

Name	Data Type	Description
<i>WorkingDirection</i> ?	enumeration	Direction from which the tool is working. Allowed values are: <i>Top</i> – From above. <i>Bottom</i> – From below.
<i>WorkingPath</i> ?	XYPair	Working path of the tool beginning at <i>@StartPosition</i> .

10.11 Cut

Cut describes one straight cut with an arbitrary tool.

Element Properties

Element referenced by: **CuttingParams, FoldingParams**

Table 10-13: Cut Element (Sheet 1 of 2)

Name	Data Type	Description
<i>CutWidth</i> ?	double	Width in points of u-shaped knife, saw blade, etc.
<i>RelativeStartPosition</i> ?	XYPair	Relative starting position of the tool. The <i>@RelativeStartPosition</i> is always based on the complete size of the input Component and not on the size of an intermediate state of the folded Sheet. The allowed value range is from 0.0 to 1.0 for each component of the XYPair, which specifies the full size of the input Component .
<i>RelativeWorkingPath</i> ?	XYPair	Relative working path of the tool beginning at <i>@RelativeStartPosition</i> . Since the tools can only work parallel to the edges, one coordinate SHALL be zero. <i>@RelativeWorkingPath</i> is always based on the complete size of the input Component and not on the size of an intermediate state of the folded Sheet. The allowed value range is from 0.0 to 1.0 for each component of the XYPair, which specifies the full size of the input Component .
<i>StartPosition</i> ?	XYPair	Starting position of the tool. If both <i>@StartPosition</i> and <i>@RelativeStartPosition</i> are specified, <i>@RelativeStartPosition</i> is ignored. At least one of <i>@StartPosition</i> or <i>@RelativeStartPosition</i> SHALL be specified.
<i>WorkingDirection</i> ?	enumeration	Direction from which the tool is working. Allowed values are: <i>Top</i> – From above. <i>Bottom</i> – From below.

Table 10-13: Cut Element (Sheet 2 of 2)

Name	Data Type	Description
<i>WorkingPath</i> ?	XYPair	Working path of the tool beginning at <i>@StartPosition</i> . Since the tools can only work parallel to the edges, one coordinate SHALL be zero. If both <i>@WorkingPath</i> and <i>@RelativeWorkingPath</i> are specified, <i>@RelativeWorkingPath</i> is ignored. At least one of <i>@WorkingPath</i> or <i>@RelativeWorkingPath</i> SHALL be specified.

10.12 Disjointing

The Disjointing Element describes how individual components are separated from one another on a stack.

Element Properties

Element referenced by:: **Component**, **DigitalPrintingParams**, **GatheringParams**, **StackingParams**

Table 10-14: Disjointing Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Number</i> ?	integer	Number of Sheets that make up one component. The <i>@OffsetUnits</i> attribute specifies the type of the component. See <i>@OffsetUnits</i> for more details.
<i>Offset</i> ?	XYPair	Offset dimension in X and Y dimensions that separates the components.
<i>OffsetAmount</i> ?	integer	The number of components that are shifted in <i>@OffsetDirection</i> simultaneously. The <i>@OffsetUnits</i> attribute specifies the type of the component counted by this attribute. See <i>@OffsetUnits</i> for more details.
<i>OffsetDirection</i> ?	enumeration	<p>Offset-shift action for the first component. A component can be offset to one of two positions—left or right.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Alternate</i> – The position of the first component of a new job is opposite to the position of the previous component and subsequent components are each offset to alternating positions. For example, if the last item in the stack was positioned to the right then the subsequent items will be positioned to the left, right, left, right and so on. <i>Left</i> – The first component of a new job is on the left, and subsequent components are each offset to alternating positions. <i>None</i> – Do not offset consecutive components. The position of all components is the same as the position of the previous component. <i>Right</i> – The first component of a new job is on the right, and subsequent components are each offset to alternating positions.

Table 10-14: Disjointing Element (Sheet 2 of 2)

Name	Data Type	Description
<i>OffsetUnits</i> ?	NMOKEN	<p>This attribute specifies the type of the component counted by the <i>@OffsetAmount</i> attribute. If <i>@Number</i> is present, it specifies the number of Sheets that make up a component (e.g., if <i>@OffsetUnits</i> is "Sets", the value of <i>@Number</i> specifies the number of sheets in a Set, and <i>@OffsetAmount</i> specifies the number of Sets). If <i>@Number</i> is not specified, it is assumed that the system has an internal way to keep track of component boundaries, whatever they may be (e.g., Set or Document boundaries). In a simple, non-VDP workflow, the product of <i>@Number</i> and <i>@OffsetAmount</i> is the number of sheets between shifts or separators.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>DocCopies</i> – every individual document is counted. <i>Docs</i> – all copies of identical documents are counted as one. <i>Jobs</i> – entire Jobs are counted. <i>SetCopies</i> – every individual Set is counted. <i>Sets</i> – all copies of identical sets are counted as one. <i>Sheets</i> – each sheet is counted. <i>Stacks</i> – each stack is counted.
IdentificationField *	element	Marks that identify the range of Sheets to be used in a Process. A scanner will scan the Sheets and detect a component boundary by scanning a mark (e.g., a bar code) that matches the description in the IdentificationField Element.
<i>InsertSheet</i> ?	element	Some kind of physical marker (e.g., a paper strip or a yellow paper Sheet) that separates the components.

10.13 FitPolicy

This Element specifies how to fit content into a receiving container (e.g., a page onto a ContentObject of an imposed sheet). See the description of each reference to **FitPolicy** to determine what the context-specific “content” is and what the “receiving container” is.

Element Properties

Element referenced by: **ImageSetterParams**, **InterpretingParams**, **Layout/ContentObject**, **Layout/MarkObject**, **RasterReadingParams**

Table 10-15: FitPolicy Element (Sheet 1 of 2)

Name	Data Type	Description
<i>ClipOffset</i> ?	XYPair	Defines the offset (position) of the imaged area in the non-rotated source image when <i>@SizePolicy</i> is "ClipToMaxPage". The values 0.0 0.0 mean that the imaged area starts at the lower left point of the receiving container. If absent, the imaged area is taken from the center of the source image.

Table 10-15: FitPolicy Element (Sheet 2 of 2)

Name	Data Type	Description
<i>GutterPolicy</i> ?	enumeration	<p>Allows printing of NUp grids even if the media size does not match the requirements of the data. <i>@GutterPolicy</i> SHALL NOT be specified when FitPolicy is referenced from a Layout Resource.</p> <p>Allowed values are:</p> <p><i>Distribute</i> – The gutters can grow or shrink to the value specified in <i>@MinGutter</i>.</p> <p><i>Fixed</i> – The gutters are fixed.</p>
<i>MinGutter</i> ?	XYPair	<p>Minimum width in points of the horizontal and vertical gutters formed between rows and columns of pages of a multi-up Sheet layout.</p> <p>The first value specifies the minimum width of all horizontal gutters and the second value specifies the minimum width of all vertical gutters.</p> <p><i>@MinGutter</i> SHALL NOT be specified when FitPolicy is referenced from a Layout resource.</p>
<i>RotatePolicy</i> ?	enumeration	<p>Specifies the policy for the Device to automatically rotate the content to optimize the fit of the content to the receiving container.</p> <p>Allowed values are:</p> <p><i>NoRotate</i> – Do not rotate.</p> <p><i>RotateOrthogonal</i> – Rotate by 90° in either direction.</p> <p><i>RotateClockwise</i> – Rotate clockwise by 90°.</p> <p><i>RotateCounterClockwise</i> – Rotate counterclockwise by 90°.</p>
<i>SizePolicy</i> ?	enumeration	<p>Allows printing even if the container size does not match the requirements of the data.</p> <p>Allowed values are:</p> <p><i>ClipToMaxPage</i> – The page contents are to be clipped to the size of the container. The printed area is either centered in the source image if no <i>@ClipOffset</i> key is given, or from that position which is determined by <i>@ClipOffset</i>.</p> <p><i>Abort</i> – Emit an error and abort printing.</p> <p><i>FitToPage</i> – The page contents are to be scaled up or down to fit the container. The aspect ratio is maintained.</p> <p><i>ReduceToFit</i> – The page contents are to be scaled down but not scaled up to fit the container. The aspect ratio is maintained.</p> <p><i>Tile</i> – the page contents are to be split into several tiles, each printed on its own surface.</p>

10.14 Fold

Fold describes an individual folding operation of the **Component**.

Element Properties

Element referenced by: **FoldingIntent**, **BinderySignature**, **FoldingParams**, **Layout**/

Table 10-16: Fold Element

Name	Data Type	Description
<i>From</i>	enumeration	Edge from which the page is folded. Allowed values are: <i>Front</i> <i>Left</i>
<i>RelativeTravel?</i>	double	Relative distance of the reference edge relative to <i>@From</i> in the coordinates of the incoming Component . The <i>@RelativeTravel</i> is always based on the complete size of the input Component and not on the size of an intermediate state of the folded Sheet. The allowed value range is from 0.0 to 1.0, which specifies the full length of the input Component . At least one of <i>@Travel</i> or <i>@RelativeTravel</i> SHALL be specified.
<i>To</i>	enumeration	Direction in which it is folded. Allowed values are: <i>Up</i> – Upwards; corresponds to a valley fold with the left/bottom side coming over the opposite side. <i>Down</i> – Downwards; corresponds to a mountain or peak fold with the left/bottom side coming under the opposite side.
<i>Travel?</i>	double	Distance of the reference edge relative to <i>@From</i> . If both <i>@Travel</i> and <i>@RelativeTravel</i> are specified, <i>@RelativeTravel</i> is ignored. At least one of <i>@Travel</i> or <i>@RelativeTravel</i> SHALL be specified.

10.15 Glue

This element provides the information to determine where and how to apply glue.

All positions and paths are specified relative to the center of the glue application tool.

Element Properties

Element referenced by: **InsertingIntent**, **BoxFoldingParams**, **CaseMakingParams**, **EndSheetGluingParams/EndSheet**, **GluingParams/GlueLine**, **HeadBandApplicationParams**, **InsertingParams**, **ThreadSewingParams**, **MediaLayers**,

Table 10-17: Glue Element (Sheet 1 of 3)

Name	Data Type	Description
<i>AreaGlue?</i>	boolean	Specifies that this GlueLine SHOULD cover the complete width of the Component it is applied to.
<i>GlueLineWidth?</i>	double	Width of the glue line in points. If not specified, the default behavior depends on the value of <i>@AreaGlue</i> : If <i>@AreaGlue = "true"</i> , then the implied width is the width of the Component . If <i>@AreaGlue = "false"</i> , then the implied width is the system dependent glue line width.
<i>GlueRef?</i>	IDREF	Reference to a MiscConsumable that represents the physical glue.

Table 10-17: Glue Element (Sheet 2 of 3)

Name	Data Type	Description
<i>GlueType</i> ?	enumeration	<p>Glue type.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>ColdGlue</i> – Any type of glue that needs no heat treatment. <i>Hotmelt</i> – Hotmelt EVA (Ethyl-Vinyl-Acetate-Copolymere) <i>Permanent</i> – Any glue that is designed not to be removed. <i>PUR</i> – Polyurethane <i>Removable</i> – Any glue that is designed to be removed.
<i>GluingPattern</i> ?	NumberList	<p>Glue line pattern defined by the length of a glue line segment (1st Element, 3rd and all odd elements of the NumberList) and glue line gap (2nd Element, 4th and all even elements of the NumberList). A solid line is expressed by the pattern (1 0).</p> <p><i>@GluingPattern</i> SHALL contain an even number of entries. If the total length of <i>@GluingPattern</i> is less than <i>@WorkingPath</i> or the length implied by <i>@RelativeWorkingPath</i>, the pattern restarts after the last gap. If the total length of <i>@GluingPattern</i> is larger than <i>@WorkingPath</i> or the length implied by <i>@RelativeWorkingPath</i>, the pattern is clipped at the end.</p>
<i>GluingTechnique</i> ?	enumeration	<p>When Glue is specified in the context of hard cover binding, then <i>@GluingTechnique</i> specifies the technique of gluing operation.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>SpineGluing</i> <i>SideGluingFront</i> <i>SideGluingBack</i>
<i>MeltingTemperature</i> ?	integer	<p>Temperature needed for melting the glue, in degrees centigrade.</p> <p>Used only when <i>@GlueType</i> = "Hotmelt" or <i>@GlueType</i> = "PUR".</p>
<i>RelativeStartPosition</i> ?	XYPair	<p>Relative starting position of the tool. The <i>@RelativeStartPosition</i> is always based on the complete size of the input Component and not on the size of an intermediate state of the folded Sheet. The allowed value range is from 0.0 to 1.0 for each component of the XYPair, which specifies the full size of the input Component.</p>
<i>RelativeWorkingPath</i> ?	XYPair	<p>Relative working path of the tool beginning at <i>@RelativeStartPosition</i>. The <i>@RelativeWorkingPath</i> is always based on the complete size of the input Component and not on the size of an intermediate state of the folded Sheet. The allowed value range is from 0.0 to 1.0 for each component of the XYPair, which specifies the full size of the input Component.</p>
<i>StartPosition</i> ?	XYPair	<p>Start position of glue line. The start position is given in the coordinate system of the mother Sheet. If both <i>@StartPosition</i> and <i>@RelativeStartPosition</i> are specified, <i>@RelativeStartPosition</i> is ignored.</p>

Table 10-17: Glue Element (Sheet 3 of 3)

Name	Data Type	Description
<i>WorkingDirection</i> ?	enumeration	Direction from which the Glue should be applied to the Component . Allowed values are: <i>Back</i> – Back side of of a sheet or product <i>Bottom</i> – Bottom of a product <i>Front</i> – Front side of of a sheet or product <i>Left</i> – Left side of a product, e.g. the spine of a left bound book. <i>Right</i> – Right side of a product, e.g. the spine of a right bound book. <i>Top</i> – Top of a product
<i>WorkingPath</i> ?	XYPair	Relative working path of the gluing tool. If both <i>@WorkingPath</i> and <i>@RelativeWorkingPath</i> are specified, <i>@RelativeWorkingPath</i> is ignored.

10.16 HolePattern

The **HolePattern** Resource describes a pattern of one or more holes.

HolePattern MAY be used to describe line hole punching which generates a series of holes with identical distance (pitch) running parallel to the edge of a Web, which is mainly used to transport paper through continuous-feed printers and finishing Devices (form processing). The final product typically is a Web with two lines of holes, one at each edge of the Web. The distance between holes within each line of holes is identical (constant pitch). In case of line hole punching *@Center* applies to the initial hole and *@Extent* applies to each hole individually.

Default behavior for *@HoleCount*: For dealing with the Default case of *@HoleCount* (i.e., when not supplied), intelligent systems will take into consideration Job s like the length of the binding edge or distance of holes to the paper edges to calculate the appropriate number of holes. For production of the holes and selection/production of the matching binding Element, the “system specified” values need to match 100% between the **HoleMaking** and the binding Process for obvious reasons. In practice, if no details are specified for **HoleMaking**, they SHOULD also be absent for binding. In this case, either the operator provides the missing value when setting up the binding Device for the Job, or the Device itself needs to have some kind of automatic hole detection mechanism.

Element Properties

Element referenced by: **HoleMakingParams**

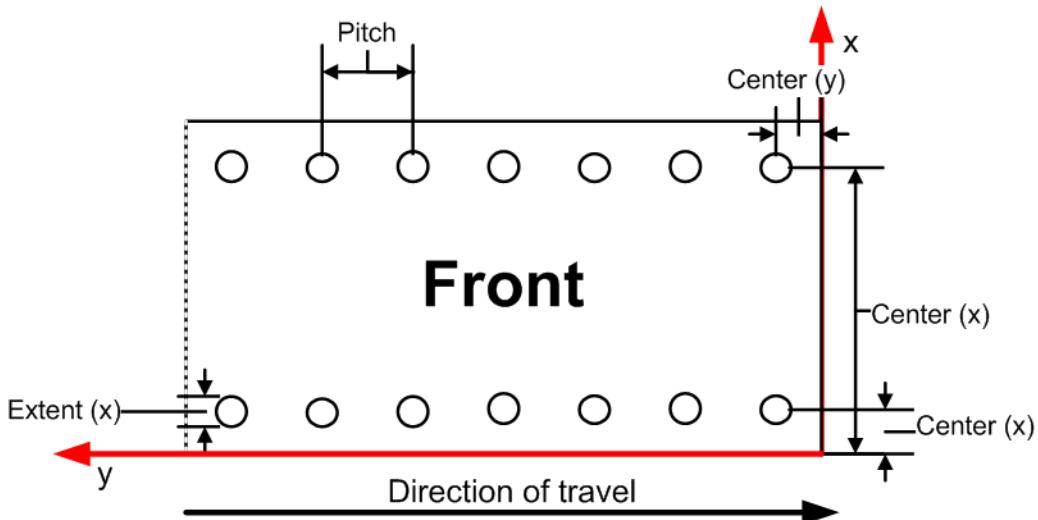
Table 10-18: HolePattern Element

Name	Data Type	Description
<i>Center</i> ?	XYPair	Position of the center of the hole relative to the Component coordinate system. For more information, see Section 6.7.2, “HoleMaking”.
<i>CenterReference</i> ?	enumeration	Defines the reference coordinate system for <i>@Center</i> . Allowed values are: <i>TrailingEdge</i> – Physical coordinate system of the component. <i>RegistrationMark</i> – The center is relative to a registration mark.
<i>Extent</i> ?	XYPair	Size (Bounding Box) of each hole, in points. If <i>@Shape</i> is “Round”, only the first entry of <i>@Extent</i> is evaluated and defines the hole diameter.

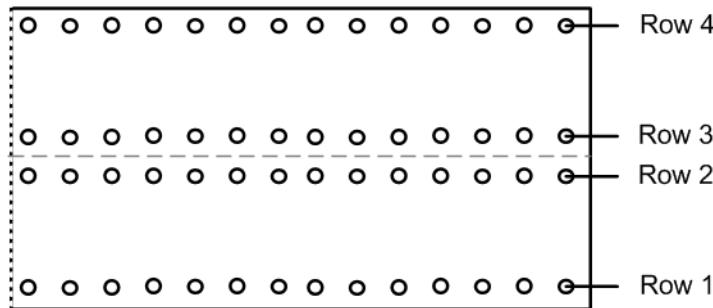
Table 10-18: HolePattern Element

Name	Data Type	Description
<i>HoleCount</i> ?	IntegerList	<p><i>@HoleCount</i> specifies the number of consecutive holes and spaces. The first entry defines the number of holes, the second entry defines the number of spaces, and consecutive entries alternately define holes (h) and spaces (s), for instance:</p> <p>"2 2 2" = "h h s s h h".</p> <p>"0 3 3 3 3" = "s s s h h h s s s h h h".</p> <p>Note: <i>@HoleCount</i> is typically applied to patterns with <i>@HoleType</i> whose enumeration values begin with a "P", "W" or "C" in Table M-1, "Naming Scheme for Hole Patterns"</p>
<i>Pattern</i> ?	NMTOKEN	<p>Prefined hole pattern.</p> <p>Allowed values are: from: Section M, "JDF XJDF/CIP4 Hole Pattern Catalog" on page 1271</p>
<i>Pitch</i> ?	XYPair	<p>If <i>@Pitch</i> is specified, this HolePattern represents a line of holes. Center-hole to center-hole distance within a line of hole and <i>@Pitch</i> represents the distance between the centers of two adjacent holes. If neither <i>@Pitch</i> nor <i>@Pattern</i> is specified, this HolePattern represents an individual hole.s.</p>
<i>Shape</i> ?	enumeration	<p>Shape of the hole.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Elliptic</i> <i>Round</i> <i>Rectangular</i>
<i>ReferenceEdge</i> ?	enumeration	<p>The edge of the Component relative to where the holes SHALL be placed.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Left</i> <i>Right</i> <i>Top</i> <i>Bottom</i> <p><i>Pattern</i> – Specifies that the reference edge implied by the value of <i>@Pattern</i> in Section M, "JDF XJDF/CIP4 Hole Pattern Catalog" on page 1271</p>

ElementElement

Figure 10-1: Hole line parameters

However, sometimes Line Hole Punching is performed for multiple webs before dividing the Web after the **HoleMaking** Process as illustrated in Figure 10-5 below:

Figure 10-2: Line hole punching for multiple webs

10.17 InsertSheet

InsertSheet Resources define Device generated images and Sheets which can be produced along with the Job. **InsertSheet** Elements include separators Sheets, error Sheets, accounting Sheets and Job Sheets. The information provided on the Sheet depends on the type of Sheet. In some cases, an **Imposition** Process can encounter **RunList** Elements that do not provide enough finished pages to complete a **Layout** Resource or its children. **InsertSheet** Resources are used to provide a standard way of completing such **Layout** Resources. **InsertSheet** Resources MAY also be used to start new Sheet Resources (e.g., to ensure that a new chapter starts on a right-hand page). In addition, **InsertSheet** MAY specify whether new media are to be inserted after the current Sheet, Signature, Instance Document or Job is completed.

InsertSheet Elements MAY be used at the beginning or end of **RunList** with a **@SheetUsage** Attribute of "Header" or "Trailer". When an **InsertSheet** appears in a **RunList**, the following precedence applies:

- 1 The **InsertSheet** with **@Usage** "FillSurface" from the **RunList** is applied first.
- 2 The **InsertSheet** with **@Usage** "FillSheet" from the **RunList** is applied.
- 3 The **InsertSheet** with **@Usage** "FillSignature" from the **RunList** is applied.

If the **RunList** of the **InsertSheet** does not supply enough content to fill a Sheet, Signature or surface, the **RunList** will be reapplied until no **PlacedObject** slots remain to be filled. When an **InsertSheet** is used in a **RunList** of a Process that does not use a **Layout** Resource (i.e., that Process is not a part of a Combined Process with **Imposition**), only **@Usage "Header"** or **"Trailer"** are valid.

Table 10-19: InsertSheet Element (Sheet 1 of 2)

Name	Data Type	Description
<i>IncludeInBundleItem</i> ?	enumeration	<p>Defines bundle items when this InsertSheet is not a Subelement of RunList. If this InsertSheet is a Subelement of a RunList, then @IncludeInBundleItem SHALL be ignored, and RunList/@EndOfBundleItem SHALL be used instead. As an example, @IncludeInBundleItem controls whether the InsertSheet is to be included in a bundle item for purposes of finishing the InsertSheet with other Sheets.</p> <p>Allowed values are:</p> <p><i>After</i> – This InsertSheet is to be included in the BundleItem that occurs after this InsertSheet. <i>"After"</i> is equivalent to <i>"None"</i> if no BundleItem is defined after this InsertSheet.</p> <p><i>Before</i> – This InsertSheet is to be included in the BundleItem that occurs before this InsertSheet. <i>"Before"</i> is equivalent to <i>"None"</i> if no BundleItem is defined before this InsertSheet.</p> <p><i>None</i> – This InsertSheet is not included in a BundleItem.</p> <p><i>New</i> – A new BundleItem is created. This InsertSheet will be in the new BundleItem by itself unless another InsertSheet with @IncludeInBundleItem = "Before" occurs immediately after this InsertSheet.</p>
<i>IsWaste</i> ?	boolean	Specifies whether the InsertSheet is waste that is to be removed from the document before further processing. If <i>"true"</i> , the InsertSheet is to be discarded when finishing the document.
<i>LayoutRef</i> ?	IDREF	Details of the Sheet that will be inserted. Contents for this Layout are drawn from the RunList included in this InsertSheet if any. If not specified, the system specified insert Sheets are used. Any InsertSheet Resources referenced by this Layout are ignored.
<i>MarkList</i> ?	NMTOKENS	<p>List of marks that are to be marked on this InsertSheet. Ignored if a Sheet is specified in this InsertSheet.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>CIELABMeasuringField</i> <i>ColorControlStrip</i> <i>ColorRegisterMark</i> <i>CutMark</i> <i>DensityMeasuringField</i> <i>IdentificationField</i> <i>JobField</i> <i>PaperPathRegisterMark</i> <i>RegisterMark</i> <i>ScavengerArea</i>

Table 10-19: InsertSheet Element (Sheet 2 of 2)

Name	Data Type	Description
<code>RunListRef?</code>	IDREF	A RunList that provides the content for the InsertSheet . Any InsertSheet Resources referenced by this RunList are ignored.
<code>SheetFormat?</code>	NMTOKEN	<p>Identifies that Device-dependent information is to be included on the InsertSheet.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Blank</i> <i>Brief</i> <i>Duplicate</i> – Valid for <code>@SheetUsage = "Interleaved"</code> or <code>"InterleavedBefore"</code>. Specifies that the interleaved Sheet is to contain the same (duplicate) content as the previous (<code>"Interleaved"</code>) or following (<code>"InterleavedBefore"</code>) Sheet. If there is content on both sides of the previous or following Sheet (duplex), then the InsertSheet has both sides duplicated. <i>Full</i> <i>Standard</i>
<code>SheetType</code>	enumeration	<p>Identifies the type of Sheet.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>AccountingSheet</i> – A Sheet that reports accounting information for the Job. <i>ErrorSheet</i> – A Sheet that reports errors for the Job. <i>FillSheet</i> – A Sheet that fills ContentObject Elements with no matching entry in the content RunList. <i>InsertSheet</i> – A Sheet that is inserted to the Job (e.g., a preprinted cover). <i>JobSheet</i> – A Sheet that delimits the Job. <i>SeparatorSheet</i> – A Sheet that delimits pages, sections, copies or Instance Documents of the Job.
<code>SheetUsage</code>	enumeration	<p>Indicates where this InsertSheet is to be produced and inserted into the set of output pages.</p> <p>Allowed values are from: Table 10-28, “SheetUsage Attribute Values”.</p>

— Attribute: `SheetUsage`

Table 10-20: SheetUsage Attribute Values (Sheet 1 of 3)

Value	Description
<code>FillForceBack</code>	Valid for <code>@SheetType = "FillSheet"</code> . Contents of the RunList of the InsertSheet are used to fill the next Finished front Page of the current Sheet before forcing the next page of the content RunList to the next Finished back Page if not already on a Finished back Page.

Table 10-20: SheetUsage Attribute Values (Sheet 2 of 3)

Value	Description
<i>FillForceFront</i>	Valid for <code>@SheetType = "FillSheet"</code> . Contents of the RunList of the InsertSheet are used to fill the next Finished back Page of the current Sheet before forcing the next Page of the content RunList to the next Finished front Page if not already on a Finished front Page. A typical use is to start a chapter on the front side of the Finished Page.
<i>FillSheet</i>	Valid for <code>@SheetType = "FillSheet"</code> . Contents from the RunList of the InsertSheet are used to fill the current Sheet.
<i>FillSignature</i>	Valid for <code>@SheetType = "FillSheet"</code> . Contents from the RunList of the InsertSheet are used to fill the current Signature.
<i>FillSurface</i>	Valid for <code>@SheetType = "FillSheet"</code> . Contents from the RunList of the InsertSheet are used to fill the current surface.
<i>Header</i>	Valid for <code>@SheetType = "InsertSheet", "JobSheet" or "SeparatorSheet"</code> . The Sheet is produced at the beginning of the Job (for <i>JobSheet</i>), or at the beginning of each copy of each Instance Document (for <i>SeparatorSheet</i>), or is appended before the current Sheet, Signature, layout or RunList as defined by its context. Contents for the Sheet are drawn from the RunList included in this InsertSheet Resource if one is included. If a RunList is not included, the inserted Sheet is filled with system-specified content defined by <code>@SheetType</code> .
<i>Interleaved</i>	Valid for <code>@SheetType = "SeparatorSheet"</code> . The Sheet is produced after each page (e.g., used to insert Sheets under transparencies). Contents for the Sheet are drawn from the RunList included in this InsertSheet Resource if one is included. If a RunList is not included, the inserted Sheet is filled with system-specified content defined by <code>@SheetType = "SeparatorSheet"</code> .
<i>InterleavedBefore</i>	Valid for <code>@SheetType = "SeparatorSheet"</code> . The Sheet is produced before each page (e.g., used to insert Sheets before transparencies). Contents for the Sheet are drawn from the RunList included in this InsertSheet Resource if one is included. If a RunList is not included, the inserted Sheet is filled with system-specified content defined by <code>@SheetType = "SeparatorSheet"</code> .
<i>OnError</i>	Valid for <code>@SheetType = "ErrorSheet"</code> . The Sheet is produced at the end of the Job only when an error or warning occurs.
<i>Slip</i>	Valid for <code>@SheetType = "SeparatorSheet"</code> . The Sheet is produced between each copy of each Instance Document. Contents for the Sheet are drawn from the RunList included in this InsertSheet Resource if one is included. If a RunList is not included, the inserted Sheet is filled with system-specified content defined by <code>@SheetType = "SeparatorSheet"</code> .
<i>SlipCopy</i>	Valid for <code>@SheetType = "SeparatorSheet"</code> . The Sheet is produced between each copy of the Job, which is defined to be when the complete RunList has been consumed. Contents for the Sheet are drawn from the RunList included in this InsertSheet Resource if one is included. If a RunList is not included, the inserted Sheet is filled with system-specified content defined by <code>@SheetType = "SeparatorSheet"</code> .

Table 10-20: SheetUsage Attribute Values (Sheet 3 of 3)

Value	Description
<i>Trailer</i>	Valid for <code>@SheetType = "AccountingSheet", "ErrorSheet", "InsertSheet", "JobSheet" and "SeparatorSheet"</code> . The Sheet is produced at the end of the Job (for <code>"AccountingSheet"</code> , <code>"ErrorSheet"</code> and <code>"JobSheet"</code>), or at the end of each copy of each Instance Document (for <code>"SeparatorSheet"</code>), or is appended after the current Sheet, Signature, layout or RunList as defined by its context. Contents for the Sheet are drawn from the RunList included in this <code>InsertSheet</code> Resource if one is included. If a RunList is not included, the inserted Sheet is filled with system specified content defined by <code>SheetType</code> . Note: use <code>@SheetType = "ErrorSheet"</code> and <code>@SheetUsage = "Trailer"</code> to always produce a Sheet that contains error or success information even if no errors or warnings occurred.

10.18 JobField

A **JobField** is a **Mark** object that specifies the details of a Job. The **JobField** Elements are also referred to as slug lines.

Element Properties

Element referenced by: **Layout/MarkObject**, **StripMark**

Table 10-21: JobField Element

Name	Data Type	Description
<i>JobFormat</i> ?	string	A formatting string used with <code>@JobTemplate</code> to generate a string. Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.
<i>JobTemplate</i> ?	string	A list of values used with <code>@JobFormat</code> to generate a string. Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.
<i>OperatorText</i> ?	string	Text from the operator. Constraint: if <code>@JobFormat</code> and <code>@JobTemplate</code> are specified, <code>@ShowList</code> , <code>@OperatorText</code> and <code>@UserText</code> SHALL NOT be specified.
<i>ShowList</i> ?	NMTOKENS	List of elements to display in the JobField . Constraint: if <code>@JobFormat</code> and <code>@JobTemplate</code> are specified, <code>@ShowList</code> , <code>@OperatorText</code> and <code>@UserText</code> SHALL NOT be specified. Values include those from: Table I-1, “Predefined variables used in <code>@XXXTemplate</code> and <code>@ShowList</code> ” on page 1203.
<i>UserText</i> ?	string	User-defined text to output with JobField . Constraint: if <code>@JobFormat</code> and <code>@JobTemplate</code> are specified, <code>@ShowList</code> , <code>@OperatorText</code> and <code>@UserText</code> SHALL NOT be specified.

10.19 MarkColor

Definition of the separations used to fill a dynamic mark.

Element Properties

Element referenced by: FillMark, StripMark

Table 10-22: MarkColor Element

Name	Data Type	Description
<i>Name</i>	string	Name of the Separation
<i>Tint</i>	double	Value from 0 (not used) to 1 (100% tint) of the Separation specified in <i>@Name</i> .

10.20 MediaLayers

MediaLayers contains an ordered list of Subelements. Each Subelement describes an individual layer of a layered **Media** Resource. The first layer in MediaLayers is the front layer of the **Media** until the last layer, which defines the back.

Element PropertiesElement referenced by: **MediaIntent, Media****Table 10-23: MediaLayers Element**

Name	Data Type	Description
GlueLine *	element	GlueLine Resource describing a glue layer of a layered Media Resource. Each GlueLine Resource SHALL have <i>GlueLine/@AreaGlue = "true"</i> .
Media *	element	Media Resourcesdescribing a layer of a layered Media Resources.

10.21 MetadataMap

The MetadataMap in RunList allows metadata embedded in PDL files to be assigned to Partition Key values, certain RunList Attributes, or Attributes created using GeneralID. During the mapping of PDL data to the XJDF document structure (see the definition in the glossary or the discussion in the **Imposition** Process), each MetadataMap Element will be evaluated for each node (Set, Document, Page, etc.) of the PDL document structure. For XML based PDL files an XPath expression will be evaluated relative to the XML node that defines each node in the document hierarchy. For non-XML based PDLs a PDL specific mapping of the XPath to the PDL document structure is used instead and the value assignment is performed on the derived XML for the PDL file. If the path specified by the XPath does not exist in the PDL, then the associated metadata value is undefined, otherwise the metadata value will be set to the conversion of the node list to a string.

When MetadataMap is specified in the context of an **IdentificationField**, data can be extracted from the barcode represented by the **IdentificationField**.

Element PropertiesElement referenced by: **IdentificationField, RunList**

Table 10-24: MetadataMap Element (Sheet 1 of 3)

Name	Data Type	Description
<i>Context</i> ?	enumeration	<p>Specifies the node context in which the XPaths specified in this MetadataMap Element are to be evaluated.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Set</i> – evaluated relative to the current set node. <i>Document</i> – evaluated relative to the current document node. <i>SubDoc0</i> – evaluated relative to the current subdocument immediately below the Document level. <i>SubDoc1</i> – evaluated relative to the current subdocument immediately below "SubDoc0" level. <i>SubDoc2</i> – see "SubDoc1", but relative to "SubDoc1" level. <i>SubDoc3</i> – see "SubDoc1", but relative to "SubDoc2" level. <i>SubDoc4</i> – see "SubDoc1", but relative to "SubDoc3" level. <i>SubDoc5</i> – see "SubDoc1", but relative to "SubDoc4" level. <i>SubDoc6</i> – see "SubDoc1", but relative to "SubDoc5" level. <i>SubDoc7</i> – see "SubDoc1", but relative to "SubDoc6" level. <i>SubDoc8</i> – see "SubDoc1", but relative to "SubDoc7" level. <i>SubDoc9</i> – see "SubDoc1", but relative to "SubDoc8" level. <i>PagePool1</i> – evaluated relative to the current Page Pool. <i>Page</i> – evaluated relative to the current page. <i>Object</i> – evaluated for each unique object on each page.
<i>DataType</i> ?	enumeration	<p>Expected data type of the metadata value.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>PartIDKeys</i> – with this value, <i>@Name</i> SHALL match a Partition Key. <p>Allowed values are also from: GeneralID/<i>(DataType</i>.</p>

Table 10-24: MetadataMap Element (Sheet 2 of 3)

Name	Data Type	Description
<i>Name</i> ?	NMTOKEN	<p>The name of the metadata.</p> <p>If <code>@DataType = "PartIDKeys"</code>, the value of <code>@Name</code> SHALL be a <code>@PartIDKeys</code> value. See Table 3-15, “Part Element” on page 88.</p> <p>If <code>@Name = "ObjectTags"</code>, then values are added to a logical pool of tag values associated with each object being processed. This pool of object tags is referenced from: ColorSpaceConversionParams/ColorSpaceConversionOp/@ObjectTags, ScreeningParams/ScreenSelector/@ObjectTags, ObjectResolution/@ObjectTags, ColorCorrectionParams/ColorCorrectionOp/@ObjectTags.</p> <p>Otherwise, <code>@Name</code> specifies the value of an implied variable (e.g., for use in GeneralID/@IDUsage, RunList/@EndOfSet, RunList/@SetCopies, RunList/@PageCopies, or RunList/@DocCopies).</p> <p>If <code>@DataType</code> is not <code>"PartIDKeys"</code> or a RunList implied variable name (e.g., RunList/@DocCopies), then the MetadataMap Element is equivalent to explicitly defining a GeneralID Element with the value being assigned by MetadataMap/@ValueFormat. The following example counts the number of Page Elements within all DocPart Elements.</p> <pre><MetadataMap DataType="integer" Name="NumPages" ValueFormat="%d" ValueTemplate="npages"> <Expr Name="npages" Path="count(..../DocPart/Page)" /> </MetadataMap></pre> <p>If multiple MetadataMap Elements specify the same name, then the specified key has the value from the last MetadataMap Element to assign a value to that key.</p> <p>If the specified <code>@Name</code> sets the value for a <code>@PartIDKeys</code> or RunList variable, where a RunList Attribute also supplies a value (e.g., RunList/@RunTag, RunList/@DocCopies, etc.), the value supplied by the RunList Attribute shall be replaced by the value supplied by the MetadataMap.</p>
<i>ValueFormat</i> ?	string	<p>Formatting value for combining extracted values from the Expr Elements.</p> <p>Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.</p>
<i>ValueTemplate</i> ?	string	<p>Arguments for combining extracted values from the Expr Elements. The argument names SHALL match the values of Expr/@Name.</p> <p>Allowed values are from: Appendix I, “Generating strings with Format and Template” on page 1203.</p>

Table 10-24: MetadataMap Element (Sheet 3 of 3)

Name	Data Type	Description
Expr *	element	<p>Each Expr Element describes a Term expression (see Section 9.1.14, “Term” on page 1019 and Section 9, “Device Capabilities” on page 969) evaluating metadata values in the PDL. If Expr/Term is not specified, or if the Term expression returns true, then the value specified by the Expr element is assigned to the key specified by MetadataMap/@Name. Expr Elements are evaluated in the XML order specified. Expr Elements with identical @Name Attributes where a previous Expr Element with that @Name has already evaluated to true SHALL NOT be processed. If any name specified in MetadataMap/@ValueTemplate is unassigned, then the key specified by MetadataMap/@Name is undefined.</p> <p>All Expr Elements return string values. These values will be type converted as necessary during processing of @ValueFormat and @ValueTemplate (See Section I, “Generating strings with Format and Template”).</p> <p>Note: if @ValueFormat contains a constant string with no format specifiers, then it is not necessary to define any Expr Elements.</p>

10.21.1 Element: Expr

Table 10-25: Expr Element

Name	Data Type	Description
Name ?	NMTOKEN	Name of this Expr. The value (as specified by @Value or extracted from @Path) SHALL be used to evaluate the parent @ValueTemplate.
Path ?	XPath	<p>If specified, and either the value returned by the Term Element (if present) is true or no Term Element is specified, then the value specified by this path is assigned to Expr/@Name.</p> <p>If the XPath specified by @Path does not evaluate to a value such as a string or number, then this Expr Element fails and any subsequent Expr Elements are evaluated. If the XPath points to an Element, then an implied XPath text() function is executed. The value is converted into a string when returned by the Expr Element. The value returned when the XPath results in a node set is undefined.</p> <p>Constraint: exactly one of @Path or @Value SHALL be specified in an Expr Element.</p>
Value ?	string	<p>If specified, and either the value returned by the Term Element (if present) is true or no Term Element is specified, then the value of this Attribute is assigned to Expr/@Name.</p> <p>Constraint: exactly one of @Path or @Value SHALL be specified in an Expr element.</p>
Term ?	element	Evaluates one or more metadata values from the PDL, and returns a true or false result. Evaluation/@Path SHALL be specified for all Evaluation Elements in the Term hierarchy.

For PPML the XPath expression will be relative to the JOB, DOCUMENT or PAGE element. Example XPath expressions:

- “METADATA/DATUM[@key = "Gender"]” will extract the value of the Gender metadata for each **XJDF** set, document and page.
- “count(PAGE)” will count the pages within a given document (only works for **XJDF** document level Nodes).
- “count(PAGE/METADATA/DATUM[@key = "special"])” will count the number of pages that have a Special metadata defined for it.

MetadataMap may also be used to set the value of certain **RunList** Attributes. These Attributes are **@EndOfSet**, **@EndOfDocument**, **@PageCopies**, **@DocCopies** and **@SetCopies**. The values set will be instantiated as if actually present in a Partitioned **RunList** for the current page or Page Pool being processed. Care should be taken to ensure their consistency across Page Pools within a document or set.

Example 10-2: MetadataMap: Setting Attributes

This example extracts the value of the **@Copies** Attribute as specified by the **@Path**, and sets the value of **RunList/@DocCopies**.

Table 10-26: MetadataMap: Setting Attributes

Value	Description
<i>EndOfSet</i>	The last page of a set of Instance Document.
<i>EndOfDocument</i>	The last page of an Instance Document.
<i>PageCopies</i>	Number of finished page copies.
<i>DocCopies</i>	Number of Instance Document copies
<i>SetCopies</i>	Number of Instance Document Set copies.

TBD 2.x Example.

```
<RunList Class="Parameter" ID="r000004" Status="Available">
  <MetadataMap DataType="integer" Name="DocCopies" ValueFormat="%d"
    ValueTemplate="ncopies">
    <Expr Name="ncopies" Path="//record/document/@Copies"/>
    <Expr Name="ncopies" Value="1"/>
  </MetadataMap>
</RunList>
```

Example 10-3: RunList/MetadataMap

In the following example, the **MetadataMap** Element maps arbitrary tags in the document to a structural **@RunTag** Partition Key. Note that any Partition Key may be mapped. Note also that although an XPath syntax is used, this may be mapped to any hierarchical structure including but not limited to XML. Finally, note that if **Dokument/@Sektion** is a value other than **"Einband"** or **"HauptTeil"**, then the **Expr** Elements assigning values to section will all fail, resulting in **@RunTags** being undefined.

TBD 2.x Example.

```
<!--this runlist points to a structured pdl with arbitrary structural
tagging-->
<RunList Class="Parameter" ID="r000004" Status="Available">
  <MetadataMap DataType="PartIDKeys" Name="RunTags"
    ValueFormat="%s%s" ValueTemplate="sex,section">
    <!--This expression maps the value of /Dokument/Rezipient/@Sex
      to a variable "sex"-->
    <Expr Name="sex" Path="/Dokument/Rezipient/@Sex"/>
    <!--Maps all elements with /Dokument/@Sektion=Einband to Cover-->
    <Expr Name="section" Value="Cover">
      <NameEvaluation Path="/Dokument/@Sektion" RegExp="Einband"/>
    </Expr>
```

```

<!--Maps all elements with /Dokument/@Sektion=HauptTeil and >50 pages
   to BigBody-->
<Expr Name="section" Value="BigBody">
  <and>
    <NameEvaluation Path="/Dokument/@Sektion" RegExp="HauptTeil"/>
    <IntegerEvaluation Path="count(PAGE)" ValueList="51 ~ INF"/>
  </and>
</Expr>
<!--Maps all elements with /Dokument/Sektion=HauptTeil and <=50 pages
   to SmallBody-->
<Expr Name="section" Value="SmallBody">
  <and>
    <NameEvaluation Path="/Dokument/Sektion" RegExp="HauptTeil"/>
    <IntegerEvaluation Path="count(PAGE)" ValueList="0 ~ 50"/>
  </and>
</Expr>
</MetadataMap>
<LayoutElement Class="Parameter">
  <FileSpec Class="Parameter"
   MimeType="application/vnd.foobar+xml" URL="bigVariable.foo"/>
</LayoutElement>
</RunList>
<!--Layout for versioned product-->
<Layout Class="Parameter" ID="r000005" PartIDKeys="RunTags" Status="Available">
  <Layout RunTags="MaleCover">
    <MediaRef rRef="r000006">
      <Part RunTags="MaleCover"/>
    </MediaRef>
  </Layout>
  <Layout RunTags="FemaleCover">
    <MediaRef rRef="r000006">
      <Part RunTags="FemaleCover"/>
    </MediaRef>
  </Layout>
  <Layout RunTags="MaleBigBody FemaleBigBody">
    <MediaRef rRef="r000006">
      <Part RunTags="MaleBigBody MaleSmallBody FemaleBigBody
        FemaleSmallBody"/>
    </MediaRef>
  </Layout>
  <Layout RunTags="MaleSmallBody FemaleSmallBody">
    <MediaRef rRef="r000006">
      <Part RunTags="MaleBigBody MaleSmallBody FemaleBigBody
        FemaleSmallBody"/>
    </MediaRef>
  </Layout>
</Layout>
<Media Class="Consumable" ID="r000006" PartIDKeys="RunTags"
  PartUsage="Implicit" Status="Available">
  <Media RunTags="MaleCover"/>
  <Media RunTags="FemaleCover"/>
  <Media RunTags="MaleBigBody MaleSmallBody FemaleBigBody FemaleSmallBody"/>
</Media>

```

10.22 MISDetails

MISDetails is a container for MIS related information.

Element Properties

Element referenced by: AuditStatus, AuditResource, ResourceCmdParams, ResourceInfo, PipeParams, JobPhase, **NodeInfo**

Table 10-27: MISDetails Element

Name	Data Type	Description
<i>Complexity</i> ?	double	<p>Complexity of the task specified by this XJDF Node in a range from 0.0 to 1.0.</p> <p>Note: the interpretation of values is implementation dependant.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>0 . 0</i> – The job is simple and therefore reduced setup and waste or higher speeds are possible. <i>0 . 5</i> – The job is of standard complexity and therefore standard setup and waste or normal speeds are possible. <i>1 . 0</i> – The job is complex and therefore more setup and waste or lower speeds are possible.
<i>CostType</i> ?	enumeration	<p>Whether or not this MISDetails is chargeable to the customer or not.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Chargeable</i> <i>NonChargeable</i>
<i>WorkType</i> ?	enumeration	<p>Definition of the work type for this MISDetails (i.e., whether or not this MISDetails relates to originally planned work, an alteration or rework).</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Original</i> – Standard work that was originally planned for the Job. <i>Alteration</i> – Work done to accommodate change made to the Job at the request of the customer. <i>Rework</i> – Work done due to unforeseen problem with original work (bad plate, Resource damaged, etc.).
<i>WorkTypeDetails</i> ?	string	<p>Definition of the details of the work type for this MISDetails (i.e., why the work was done).</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>CustomerRequest</i> – The customer requested change(s) requiring the work. <i>EquipmentMalfunction</i> – Equipment used to produce the Resource malfunctioned; Resource needs to be created again. <i>InternalChange</i> – Change was made for production efficiency or other internal reason. <i>ResourceDamaged</i> – A Resource needs to be created again to account for a damaged Resource (damaged plate, etc.). <i>UserError</i> – Incorrect operation of equipment or incorrect creation of Resource requires creating the Resource again.

10.23 ObjectResolution

ObjectResolution defines a resolution depending on *@SourceObjects* data types.

Element Properties

Element referenced by: **InterpretingParams, RenderingParams, TrappingParams**

Table 10-28: ObjectResolution Element

Name	Data Type	Description
<i>AntiAliasing</i> ?	NMTOKEN	<p>Indicates the anti-aliasing algorithm that the Device SHALL apply to the rendered output images. An anti-aliasing algorithm causes lines and curves to appear smooth which would otherwise have a jagged appearance, especially at lower resolutions such as 300 dpi and lower.</p> <p>Values include:</p> <p><i>AntiAlias</i> – Anti-aliasing SHALL be applied. The algorithm is system specified.</p> <p><i>None</i> – Anti-aliasing SHALL NOT be applied.</p>
<i>ObjectTags</i> ?	NMTOKENS	<p>Tags associated with individual objects that this <i>ObjectResolution</i> SHALL be applied to. Each tag specified in <i>@ObjectTags</i> is logically anded with the object type(s) specified by <i>@SourceObjects</i>, enabling first qualification by object type (such as image), and then tags associated with those objects.</p> <p>The values of <i>@ObjectTags</i> depends on the PDL that the <i>ObjectResolution</i> is applied to..</p> <p><i>@ObjectTags</i> SHALL apply only to objects whose tag pool includes all the tags in the value of <i>@ObjectTags</i>.</p>
<i>Resolution</i>	XYPair	Horizontal and vertical output resolution in DPI.
<i>SourceObjects</i> ?	enumerations	<p>Identifies the class(es) of incoming graphical objects to render at the specified resolution.</p> <p>Allowed values are:</p> <p><i>All</i></p> <p><i>ImagePhotographic</i> – Contone images.</p> <p><i>ImageScreenShot</i> – Images largely comprised of rasterized vector art.</p> <p><i>LineArt</i> – Vector objects other than text.</p> <p><i>SmoothShades</i> – Gradients and blends.</p> <p><i>Text</i></p>

10.24 OCGControl

Table 10-29: OCGControl Element (Sheet 1 of 2)

Name	Data Type	Description
<i>IncludeOCG</i> ?	boolean	<p>Defines whether the optional content group(s) identified by <i>@OCGName</i> are to be included in the RunList. If "<i>true</i>", then the layer SHALL be included. If "<i>false</i>", it SHALL NOT.</p> <p>Note that the contents stream of excluded OCGs SHALL still be interpreted so that changes to CTM, etc., are acted on. The objects drawn in excluded OCGs SHALL NOT be rendered.</p>

Table 10-29: OCGControl Element (Sheet 2 of 2)

Name	Data Type	Description
<i>OCGName</i>	string	The name of the optional content group(s) that SHALL be included or excluded. Note that the <i>@Name</i> Attribute of an optional content group entry is encoded as a PDF text string, and <i>@OCGName</i> is encoded with the Unicode variant identified in the XJDF file header; names SHALL be re-encoded as necessary for comparison. Using a value for <i>@OCGName</i> that does not match any OCG in the referenced PDF file is an error, independent of the value of <i>@IncludeOCG</i> .

10.25 Perforate

Perforate describes one perforated line.

Element Properties

Element referenced by: **FoldingParams, PerforatingParams**

Table 10-30: Perforate Element (Sheet 1 of 2)

Name	Data Type	Description
<i>Depth</i> ?	double	Depth of the perforation, in microns [μm].
<i>RelativeStartPosition</i> ?	XYPair	Relative starting position of the tool. The <i>@RelativeStartPosition</i> is always based on the complete size of the input Component and not on the size of an intermediate state of the folded Sheet. The allowed value range is from 0.0 to 1.0 for each component of the XYPair, which specifies the full size of the input Component . At least one of <i>@StartPosition</i> or <i>@RelativeStartPosition</i> SHALL be specified.
<i>RelativeWorkingPath</i> ?	XYPair	Relative working path of the tool beginning at <i>@RelativeStartPosition</i> . Since the tools can only work parallel to the edges, one coordinate SHALL be zero. The <i>@RelativeWorkingPath</i> is always based on the complete size of the input Component and not on the size of an intermediate state of the folded Sheet. The allowed value range is from 0.0 to 1.0 for each component of the XYPair, which specifies the full size of the input Component . At least one of <i>@WorkingPath</i> or <i>@RelativeWorkingPath</i> SHALL be specified.
<i>StartPosition</i> ?	XYPair	Starting position of the tool. If both <i>@StartPosition</i> and <i>@RelativeStartPosition</i> are specified, <i>@RelativeStartPosition</i> is ignored. At least one of <i>@StartPosition</i> or <i>@RelativeStartPosition</i> SHALL be specified.
<i>TeethPerDimension</i> ?	double	Number of teeth in a given perforation extent in teeth/point. MicroPerforation is defined by specifying a large number of teeth ($@TeethPerDimension > 1000$).
<i>WorkingDirection</i> ?	enumeration	Direction from which the tool is working. Allowed values are: <i>Top</i> – From above. <i>Bottom</i> – From below.

Table 10-30: Perforate Element (Sheet 2 of 2)

Name	Data Type	Description
<i>WorkingPath</i> ?	XYPair	Working path of the tool beginning at <i>@StartPosition</i> . Since the tools can only work parallel to the edges, one coordinate SHALL be zero. If both <i>@WorkingPath</i> and <i>@RelativeWorkingPath</i> are specified, <i>@RelativeWorkingPath</i> is ignored. At least one of <i>@WorkingPath</i> or <i>@RelativeWorkingPath</i> SHALL be specified.

10.26 RefAnchor

RefAnchor describes the relative position with respect to a related element in a layout. Depending on the value of *@AnchorType*, it specifies either a parent Element or a sibling Element.

Element Properties

Element referenced by: **Layout/MarkObject**, **Content/PositionObj**, **StrippingParams/StripMark**

Table 10-31: RefAnchor Element

Name	Data Type	Description
<i>Anchor</i> ?	Anchor	<i>@Anchor</i> specifies the origin (0,0) of the vector specified in the rotated coordinate system of the related layout element.
<i>AnchorType</i> ?	enumeration	<p>Role of this RefAnchor.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Parent</i> – The layout element referenced by this RefAnchor is a parent. This layout element is transformed with the parent. <i>Sibling</i> – The layout element referenced by this RefAnchor is a sibling. Both layout elements share a common parent. The parent of this layout element is specified as the RefAnchor of the first child in the chain of siblings.
<i>rRef</i> ?	IDREF	<p>Reference to a layout element that this layout element is positioned relative to. If <i>@rRef</i> is not specified, the page or sheet defined by the layout element is the parent container.</p> <p><i>@rRef</i> SHALL be specified if <i>@AnchorType = "Sibling"</i>.</p>

10.27 RegisterRibbon

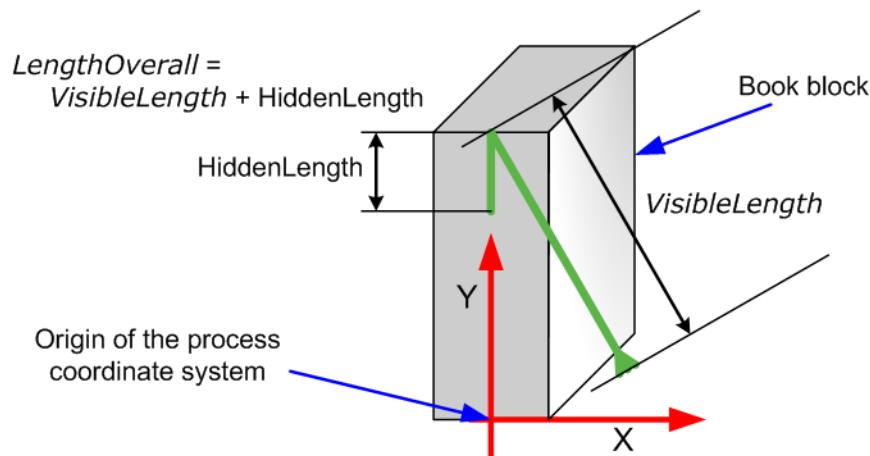
Description of register ribbons. For the register ribbon, the length SHALL be specified. There are two parameters, as shown in Figure 10-6, “RegisterRibbon lengths and coordinate system for BlockPreparation,” on page 1114:

Table 10-32: RegisterRibbon Element (Sheet 1 of 2)

Name	Data Type	Description
<i>LengthOverall</i> ?	double	Overall length of the register ribbon (i.e., <i>@VisibleLength</i> + HiddenLength in Figure 10-6). Note “HiddenLength” is not an Attribute
<i>Material</i> ?	string	Material of the register ribbon.
<i>RibbonColor</i> ?	NamedColor	Color of the ribbon.
<i>RibbonColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If <i>@RibbonColorDetails</i> is supplied, <i>@RibbonColor</i> SHOULD also be supplied.

Table 10-32: RegisterRibbon Element (Sheet 2 of 2)

Name	Data Type	Description
<i>RibbonEnd</i> ?	NMOKEN	End of the Ribbon. Values include: <i>Cut</i> <i>CutSealed</i> <i>Knot</i> <i>SealedOffset</i> – The ribbon is sealed a distance from the cut.
<i>VisibleLength</i> ?	double	Length of the register ribbon which will be seen when opening the book. See Figure 10-6.

Figure 10-3: RegisterRibbon lengths and coordinate system for BlockPreparation

10.28 ScreenSelector

Description of screening for a selection of source object types and separations.

Element Properties

Element referenced by: ColorSpaceConversionOp, ScreeningParams

Table 10-33: ScreenSelector Element (Sheet 1 of 4)

Name	Data Type	Description
<i>Angle</i> ?	double	Specifies the first angle of the screen when AM screening is used, otherwise @Angle is ignored. At most one of @Angle or @AngleMap SHALL be specified. If neither @Angle nor @AngleMap are specified, the angle is determined by the default of the selected @ScreeningFamily.

Table 10-33: ScreenSelector Element (Sheet 2 of 4)

Name	Data Type	Description
<i>AngleMap</i> ?	string	<p>Specifies the mapping of the angle of the screen to the angle of a different separation when AM screening is used. For example, a spot color that has the same screening angle as the cyan separation is specified by <i>@AngleMap = "Cyan"</i>. In FM screening, <i>@AngleMap</i> specifies the mapping of the separation specific screen functions (e.g., threshold arrays). At most one of <i>@Angle</i> or <i>@AngleMap</i> SHALL be specified. This mapping is not transitive, so, when <i>@Separation</i> already specifies a color with a known default, it specifies the angle of the separation defined by <i>@AngleMap</i> prior to that separation being mapped. Note that, in general, the known default will be a CMYK process color, but it can also be another process color (e.g., Hexachrome™). The following example specifies that "Black" is to be mapped to the "Cyan" default separation and "Cyan" to the "Black" default separation. The third line maps Spot1 to Magenta.</p> <pre><ScreenSelector AngleMap="Black" Separation="Cyan"/> <ScreenSelector AngleMap="Cyan" Separation="Black"/> <ScreenSelector AngleMap="Magenta" Separation="Spot1"/></pre>
<i>DotSize</i> ?	double	Specifies the dot size of the screen, in microns [μm], when FM screening (<i>@ScreeningType = "FM"</i> or <i>"Adaptive"</i>) is used, otherwise <i>@DotSize</i> is ignored.
<i>Frequency</i> ?	double	<p>Specifies the halftone screen frequency in lines per inch (lpi) of the screen when AM screening is used, otherwise <i>@Frequency</i> is ignored. With some screens, frequency can change as a function of gray level. In this case, the <i>@Frequency</i> value is interpreted for a midtone (50%) gray level.</p> <p>If <i>@Frequency</i> is not specified, the frequency is determined by the default of the selected <i>@ScreeningFamily</i>.</p>
<i>ObjectTags</i> ?	NMTOKENS	<p>Tags associated with individual objects that this ScreenSelector SHALL be applied to. Each tag specified in <i>@ObjectTags</i> is logically anded with the object type(s) specified by <i>@SourceObjects</i>, enabling first qualification by object type (such as image), and then tags associated with those objects.</p> <p>The values of <i>@ObjectTags</i> depends on the PDL that the ScreenSelector is applied to.</p> <p><i>@ObjectTags</i> SHALL apply only to objects whose tag pool includes all the tags in the value of <i>@ObjectTags</i>.</p>
<i>ScreeningFamily</i> ?	string	Vendor specific screening family name.
<i>ScreeningType</i> ?	enumeration	<p>General type of screening.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> <i>Adaptive</i> <i>AM</i> – Can be line or dot. See <i>@SpotFunction</i>. <i>ErrorDiffusion</i> <i>FM</i> – Includes all stochastic screening types. <i>HybridAM-FM</i> <i>HybridAMline-dot</i>

Table 10-33: ScreenSelector Element (Sheet 3 of 4)

Name	Data Type	Description
<i>Separation</i> ?	string	The name of the separation. If <i>@Separation</i> = "All", the ScreenSelector is to be applied to all separations that are not specified explicitly. Values include: All
<i>SourceFrequencyMax</i> ?	double	Specifies the maximum line frequency of screens which is to be matched from the source file when screen matching is to be done. Note that this is a filter that selects on which objects to apply this ScreenSelector.
<i>SourceFrequencyMin</i> ?	double	Specifies the minimum line frequency of screens which is to be matched from the source file when screen matching is to be done. Note that this is a filter that selects on which objects to apply this ScreenSelector.
<i>SourceObjects</i> ?	enumerations	Identifies the class(es) of incoming graphical objects on which to use the selected screen. Allowed values are: All <i>ImagePhotographic</i> – Contone images. <i>ImageScreenShot</i> – Images largely comprised of rasterized vector art. <i>LineArt</i> – Vector objects other than text. <i>SmoothShades</i> – Gradients and blends. <i>Text</i>

Table 10-33: ScreenSelector Element (Sheet 4 of 4)

Name	Data Type	Description
<i>SpotFunction</i> ?	NMTOKEN	<p>Specifies the spot function of the screen when AM screening is used. In general, it is common for a spot function to change its shape as a function of gray level. Response to these spot function names MAY be implementation-dependent. These example names are the same as the spot function names defined in PDF.</p> <p>Values include:</p> <ul style="list-style-type: none"> <i>Round</i> <i>Diamond</i> <i>Ellipse</i> <i>EllipseA</i> <i>InvertedEllipseA</i> <i>EllipseB</i> <i>EllipseC</i> <i>InvertedEllipseC</i> <i>Line</i> <i>LineX</i> <i>LineY</i> <i>Square</i> <i>Cross</i> <i>Rhomboid</i> <i>DoubleDot</i> <i>InvertedDoubleDot</i> <i>SimpleDot</i> <i>InvertedSimpleDot</i> <i>CosineDot</i> <i>Double</i> <i>InvertedDouble</i>

Chapter 11 Building a System Around XJDF

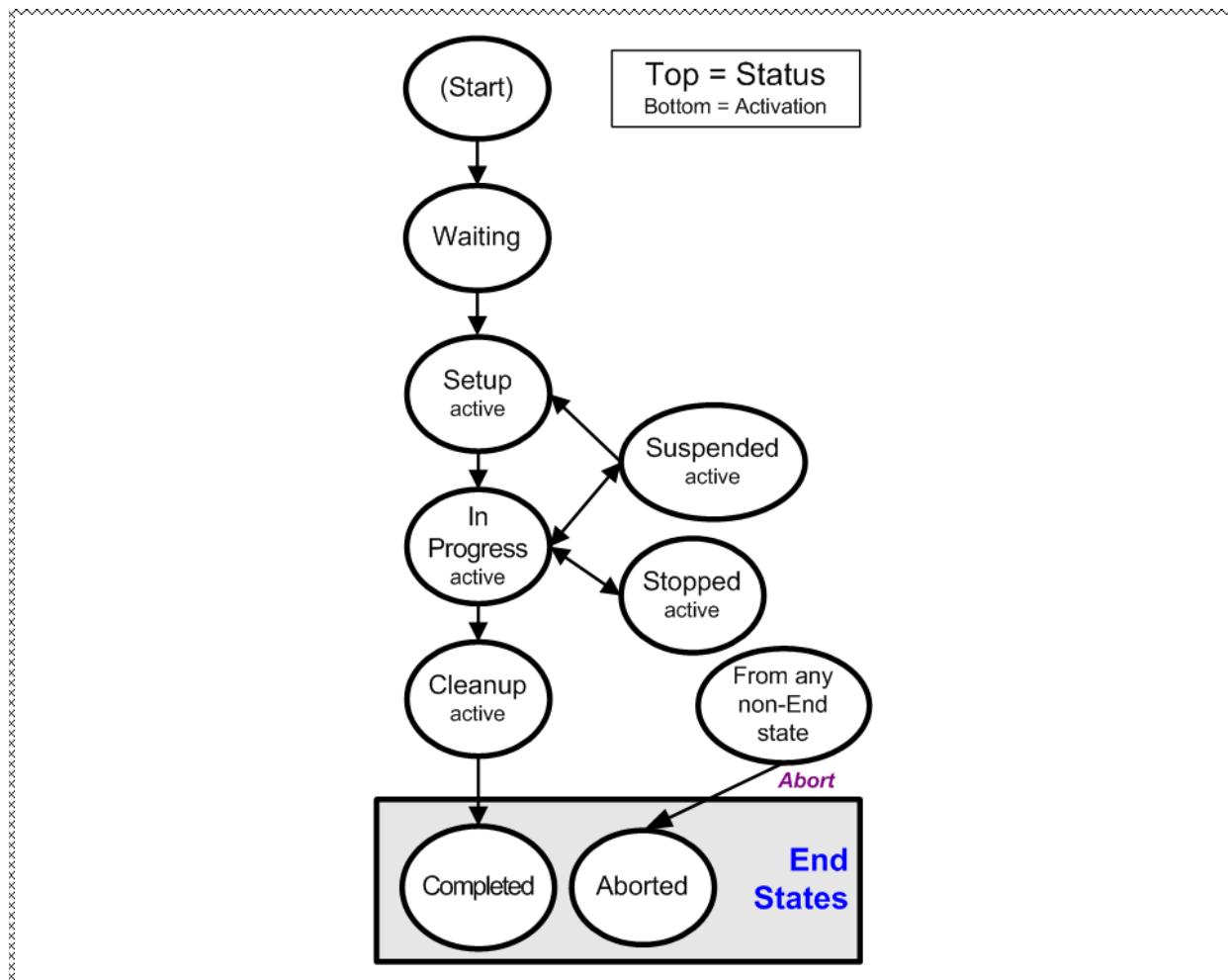
11.1 Implementation Considerations and Guidelines

A Device SHALL be able to consume the inputs and produce the outputs for each Process type it is able to execute.

11.2 Status Transitions

A process that is represented by an XJDF will go through various states during its life time as is described in Figure 11-1, “Life Cycle of a Process”. Note that the Node NEED NOT go through all phases such as Setup or Cleanup explicitly if the device does not physically support these phases. In this case the phases as described in Figure 11-1, “Life Cycle of a Process” SHALL be skipped and processing SHALL continue with the next supported phase.

Figure 11-1: Life Cycle of a Process



11.3 Execution Model

XJDF provides a range of options that help Controllers tailor a processing system to the needs of the workflow and of the Job itself. The following sections explain the ways in which Devices execute processes using these various options.

Modification note: whereas JDF 1.x supported explicit encoding of process networks, XJDF assumes that the network is implemented in a proprietary fashion by the Controllers. Each individual XJDF Node therefore contains only information that is specific to the receiving Device. Nonetheless, XJDF enables the Execution models detailed below.

The processing model of **XJDF** is based on a producer/consumer model. Devices that process **XJDF** act both as producers and consumers of Resources.

11.3.1 Serial Processing

The simplest Process sequence is serial processing. In serial processing, the Controller sends an XJDF to a Device and waits until the first XJDF has been completely processed before sending a subsequent XJDF belonging to the same Job to the same or a different Device.

11.3.2 Partial Processing of Nodes with Partitioned Resources

Some processes apply to multiple parts such as multiple sheets or plates. The partitioning structure of the **NodeInfo** resource SHALL define the sequencing of the process steps. The Device SHALL process all work steps that are explicitly specified in **NodeInfo** partitions.

If a Device is only capable of performing more granular worksteps than **NodeInfo** requires, the Device MAY split the execution into multiple work steps. For Instance, a non-perfecting press MAY process a request for a duplex sheet in two runs - one for front and one for back.

11.3.3 Parallel Processing

Some processes are independent of one another and therefore it is possible to execute them in parallel. Examples are prepress of individual pages prior to impositioning or printing of individual sheets prior to binding. In parallel processing, the Controller sends an **XJDF** to one Device and does not wait until the first **XJDF** has been processed before sending a subsequent **XJDF** belonging to the same Job to a different Device or to the same device if it is capable of processing multiple work steps simultaneously, e.g. a multi-threaded RIP.

11.3.4 Overlapping Processing

Some processes, e.g. a long print run, take a long time to complete while constantly creating intermediate output that is already available for processing by a subsequent process prior to completion of the initial process. In Overlapping processing, the Controller sends an **XJDF** to the initial Device and does not wait until the first **XJDF** has been processed before sending a subsequent **XJDF** that describes the next process step of the same job to a different Device. The subsequent Device MAY begin processing as soon as the initial Device has produced sufficient resources for the subsequent Device. The communication between the Devices SHOULD use PipeControl **XJMF** messages as described in Section 11.3.4, “Overlapping Processing”. If this is technically not feasible, out of bands communication such as a pallet of printed paper that was delivered by a fork lift MAY be used as process control.

ResourceSet/Dependent provides information that is required to setup a Pipe. Any Resource MAY be transformed into a pipe Resource by specifying **Dependent/@PipeProtocol**.

The two following diagrams show the ways in which the Process producing the Resource and the Process consuming the Resource SHALL be synchronized.

Dependent/@JMFURL – @JMFURL SHALL specify the URL that receives CommandPipeControl **XJMF** messages.

Dependent/@PipePartIDKeys – sPipePartIDKeys SHALL specify the granularity of a pipe request for partitioned resources.

Dependent/@PipePause – specifies at which resource level to pause a pipe.

Dependent@*PipeProtocol* – specified the protocol that SHALL be used to control the pipe.

Dependent@*PipeResume* – specifies at which resource level to resume a pipe.

Figure 11-4, “Example of status transitions in case of overlapping Processing” gives an example of a view on the dynamics of a pipe Resource. The available level of the pipe Resource, represented as R2, and the availability status of two Resources, represented as R1 and R3, are changing along a time line. Below the progressions of these Resources is the **NodeInfo/@NodeStatus** of two Processes — P1 and P2. P1 represents the Process producing the pipe Resource and P2 represents the Process consuming that Resource.

Figure 11-2: Example of a Pipe Resource Linking Two Processes via Pull

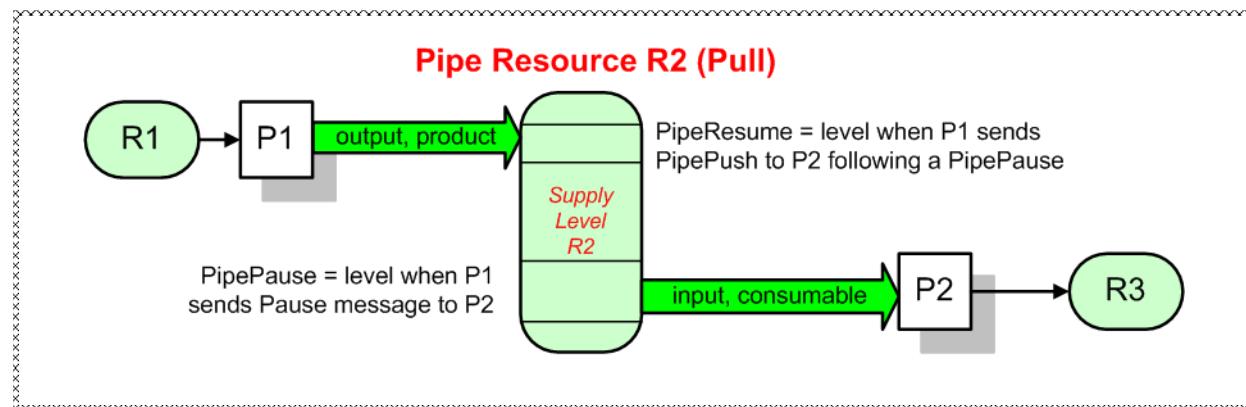


Figure 11-3: Example of a Pipe Resource Linking Two Processes via Push

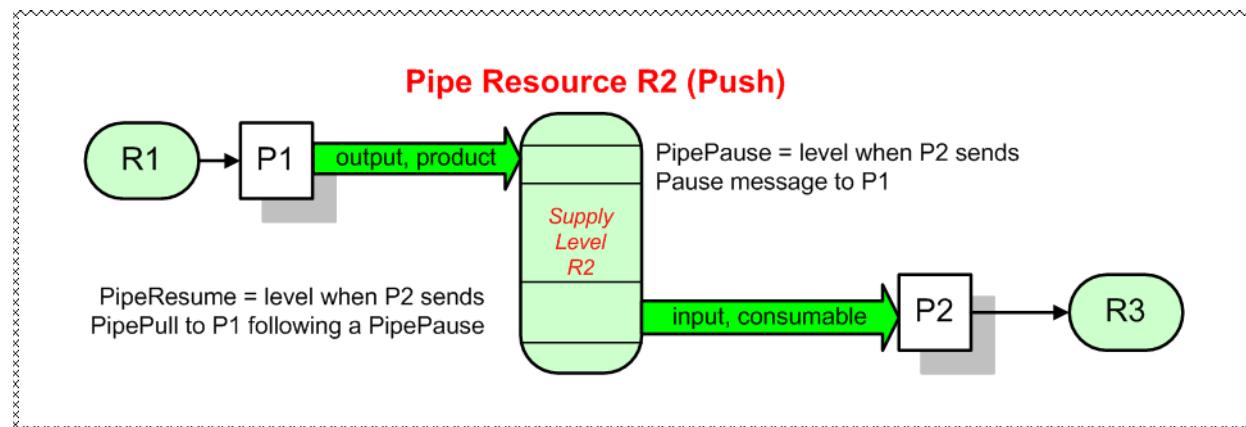
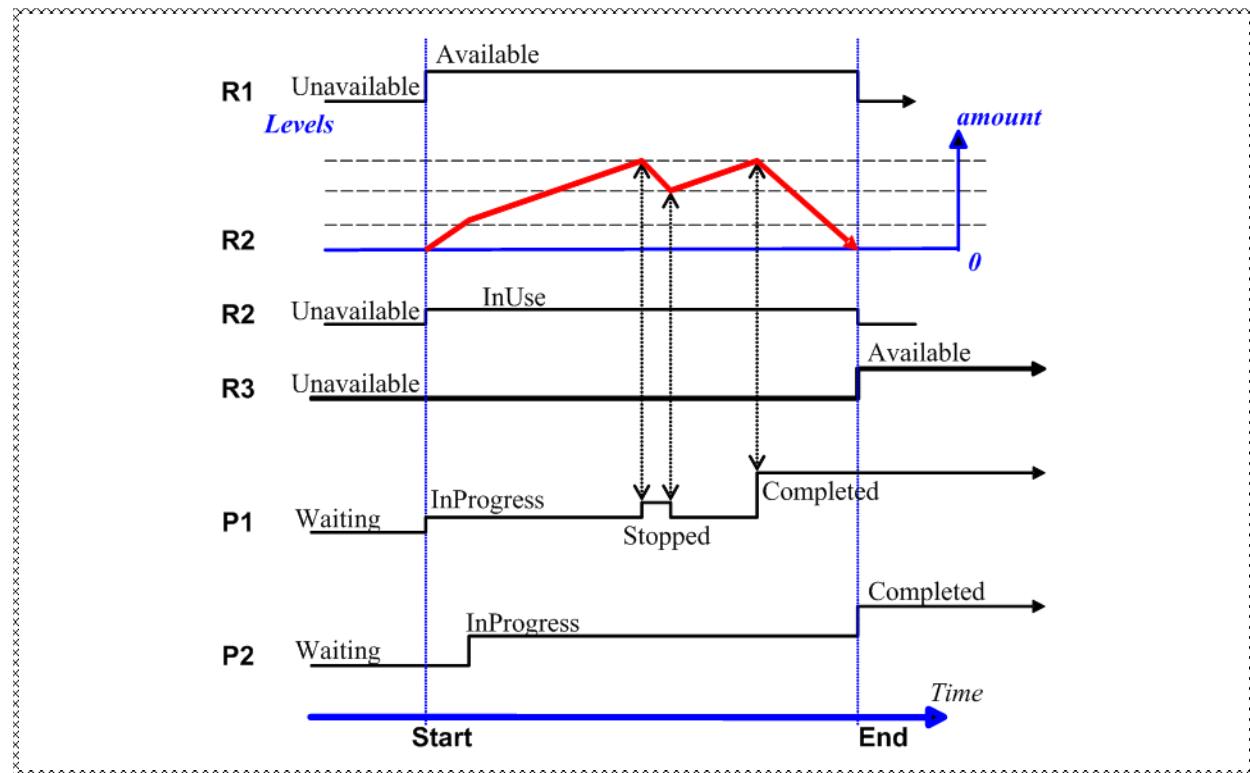


Figure 11-4: Example of status transitions in case of overlapping Processing

11.3.4.1 Dynamic Pipes

In addition to abstractly declaring pipe properties, **XJMF** provides pipe Messages that allow dynamic control of pipes. Dynamic pipes can be used to model situations where the amount of Resources is not known beforehand but becomes known during processing. An example of this behavior is a long press run where new plates are needed during a press run because of quality deterioration. The exact point in time where quality becomes unacceptable is not predetermined and might even vary from separation to separation. Dynamic pipes provide the flexibility to adjust to changing situations of this nature.

Another usage of dynamic Pipes is linking the output of a variable data print job to various components. Examples include a Pipe describing the **RunList** that links the RIP to a print engine or a Pipe describing the **Component** that links the printer to finishing equipment or individual finishing devices. In this case, the **RunList** and **Component** are templates that are logically expanded in increments by the Pipe messages.

Dynamic pipes provide a **@PipeURL** Attribute that allows dynamic requests for a status change of the pipe while a Process is executing. Dynamic requests use **XJMF** pipe control Messages (see Section 5.7, “Messages for Pipe Control”) sent to another Controller whose URL address is specified by the **@PipeURL** Attribute of the respective **Dependent**. Depending on the values of the **Dependent/@PipeProtocol** Attribute, the following actions are possible.

- “**JMFPull**”: The consumer initiates the pipe by sending an **@Operation = Pull** Message to its **@PipeURL**. The consumer MAY request new resources by sending **@Operation = Pull** Messages. If the producer reaches the pipe-pause (low water) mark, or is incapable of fulfilling **@Operation = Pull** Messages for other reasons such as a malfunction, it SHOULD send an **@Operation = Pause** Message to the consumer. Once it has reached the pipe-resume (high water) mark, or the malfunction has been removed, it SHOULD send an **@Operation = Push** Message to the consumer to inform the consumer that it can commence sending **@Operation = Pull** Messages. The consumer SHOULD send an **@Operation = Close** Message to the producer if the consumer does not require any further Resources.

- "*JMFPush*": The producer initiates the pipe by sending an @Operation = Push Message to its @PipeURL. The producer MAY dispatch new resources by sending @Operation = Push messages. If the consumer reaches the pipe-pause (high water) mark, or is incapable of fulfilling @Operation = Push requests for other reasons such as a malfunction, it SHOULD send an @Operation = Pause message to the producer. Once it has reached the pipe-resume mark (low water), or the malfunction has been removed, it SHOULD send an @Operation = Pull to the producer to inform the producer that it can commence sending @Operation = Push messages. The producer SHOULD send an @Operation = Close Message to the consumer if the producer cannot provide any further Resources.

When dynamic pipes are used, @PipeResume and @PipePause define the buffering parameters that lead to a Pipe control message to the remote device. The pipe control Messages described later in Section 5.7, "Messages for Pipe Control" are designed to establish communication between Processes at both ends of dynamic pipe, even if the corresponding Processes are spawned separately.

The following table summarizes the actions to be taken when the buffer in a dynamic pipe reaches a certain level "L".

Table 11-1: Actions generated when a dynamic-pipe buffer passes various levels

PipeProtocol (sender)	Situation	Message	Description
JMFPull (consumer)	ready to process	@Operation = Pull	The consumer has processing or buffering available and therefore the producer SHOULD produce resources.
JMFPull (creator)	L < @PipePause	@Operation = Pause	The creator has no resources available and therefore the consumer SHALL refrain from sending @Operation = Pull messages.
JMFPull (creator)	L > @PipeResume	@Operation = Push	Sufficient Resources have been produced by the creator and are ready for delivery to the consumer.
JMFPush (producer)	resources available	@Operation = Push	Sufficient resources have been produced by the creator and are ready for delivery to the consumer, therefore the consumer SHOULD consume resources.
JMFPush (consumer)	L > @PipePause	@Operation = Pause	The consumer has no processing or buffering space available and therefore the creator SHALL refrain from sending @Operation = Push messages.
JMFPush (consumer)	L < @PipeResume	@Operation = Pull	The consumer has processing or buffering available and therefore the producer SHOULD commence sending @Operation = Push Messages.

Dynamic pipes are initially dormant and SHALL be activated by an explicit request. If Resource/@PipeProtocol = "JMF", dynamic pipe requests MAY be initiated by both ends of the pipe. As soon as the Pipe has been initiated, actions that are required by the implied @PipeProtocol ("JMFPush" or "JMFPull") SHALL be applied. For example, a print Process might notify an off-line finishing Process when a certain amount is ready by sending a @Operation = Push Message, or the printing Process might request a new plate by sending a @Operation = Pull Message.

11.3.4.2 Example JMFPush Sequence

This section illustrates the concept of dynamic pipes using the example of variable data near line finishing being controlled by a variable data digital press.

The exchange resource is a **Component** that is the output of the **DigitalPrinting** Combined Node and the input of a Combined **Folding** and **Stitching** booklet maker. The actual **XJMF** messages are provided in Section 0.5.3, “JMF XJMF Pipe Messages” on page 1308 %%with example digital finishing.

Table 11-2: Event Sequence in Digital Finishing

Direction	Type	PipeParams Description
P->C	@Operation = PUSH Sheet 0/1; Cover; Set 0	Initialize Pipe
P->C	@Operation = Push Sheet 0/5; Set 0	Next sheet
P->C	@Operation = Push	Lots of next sheets
P->C	@Operation = Push Sheet 4/7; Body; Set 35	Next sheet
P-<-C	@Operation = Pause Sheet 4/7; Set 35	Paper Jam in finisher - destroys Set 34 and 35
P-<-C	@Operation = Pull Sheet 0; Set 34	Restart at Page 0 of Doc 34
P->C	@Operation = Push Sheet 0/3; Set 34	Restart Pipe
P->C	@Operation = Push	Lots of next sheets
P->C	@Operation = Pause Sheet 4/7; Cover; Set 122	Paper Jam in printer - destroys Set 122
P-<-C	@Operation = Pull Sheet 0; Set 122	Restart at Page 0 of Doc 34(optional)
P->C	@Operation = Push Sheet 0; Set 122	Restart at Page 0 of Doc 122
P->C	@Operation = Push	Lots of next sheets
P->C	@Operation = Push Sheet 2/3; Body; Set 221	Last sheet - note zero based counting for index
P->C	@Operation = Close	Done

11.3.4.3 Comparison of Non-Dynamic and Dynamic Pipes

The **Dependent** Element between non-dynamic pipes provides the buffering parameters for the Process to which the **Dependent** Element belongs. Therefore, many Processes can link to the same pipe Resource. Furthermore, each Process has its own buffering parameters, whether it is a consumer or a producer. In order to control non-dynamic pipes, one master Controller SHALL control all Processes linked to the pipe Resource.

In contrast, dynamic pipes provide a URL address to control a Process at the other pipe end. Then the buffering parameters of the **Dependent** Element control the Process at the other end. In the case of dynamic pipes, no master Controller is needed to control the pipe. Control is accomplished by sending pipe Messages. If pipe Resources are linked to multiple consumers or producers, such as two finishing lines that consume the output of one press one palette at a time, it is up to implementation to ensure consistency of the Processes.

11.3.4.4 Metadata in Pipe Messages

TBD

11.3.5 Approval, Proofing, Quality Control and Verification

In many cases, it is desirable to ensure that an executed Process or set of Processes have been executed completely and/or correctly. In the graphic arts industry this is often accomplished by generating proofs and signing approvals. **XJDF** defines the approval Process and the verification Processes by allowing an OPTIONAL **ApprovalDetails** Input ResourceSet in any Process.

The **Approval**, **QualityControl** and **Verification** Processes accept any Resource as input and output that same Resource along with **ApprovalDetails** Resource if approved. For hard copy proofing, a **DigitalPrinting** Process) generates the hard proof which is input to an **Approval**. For soft proofing, a **Rendering** or **PDLCreation** Process generates the soft proof which is input to an **Approval** Process.

XJDF provides a **QualityControl** Process to verify that the output of a Process fulfills certain quality criteria. **QualityControl** differs from the **Verification** Process, which verifies the completeness of a given set of Resources.

11.3.6 Error Handling

Error handling is an implementation-dependent feature of **XJDF**-based systems. The **AuditPool** Element provides a container where errors that occur during the execution of a **JDF** Node are to be logged using **Notification** Elements. **Notification** Elements MAY also be sent in **XJMF** messages. The content of the **Notification** Element is described in Table 5.5.6.3.1, “Notification” on page 227. For a list of predefined error codes, see Appendix D, “Supported Error Codes in XJMF and Notification Elements” on page 1183. Further details about error handling are provided in the next four sections.

11.3.7 Classification of Notifications

AuditNotification/Notification Elements are classified by the **@Class** Attribute. Every workflow implementation SHALL associate a **@Class** with all events on an event-by-event basis. For values, see **AuditNotification/Notification/@Class** in Table 5.5.6.3.1, “Notification” on page 227.

11.3.8 Event Description

A description of the event is given by a generic **Comment** Element, which is REQUIRED for the **Notification** Classes *“Information”*, *“Warning”*, *“Error”* or *“Fatal”*. For example, after a Process is aborted, error information describing a Device error MAY be logged in the **Comment** Element of the **Notification** Element. If phase times are logged, the **AuditStatus** Element that logged the transition to the *“Aborted”* state MAY also contain a local **Comment** Element that describes the cause of the Process abortion. **AuditStatus** and **Notification** Elements are OPTIONAL Subelements of the **AuditPool**, which is described in Section 3.7, “AuditPool and Audit”.

11.3.9 Error Logging in the JDF File

An **XJDF**-compliant Device SHOULD log an error by inserting an **AuditPool/AuditNotification**.

11.3.10 Error Handling via Messaging (XJMF)

An **XJMF** with a **Notification** Element in the Message body SHOULD be sent through all persistent channels that subscribed events of Class *“Error”*. In order to receive Notifications, **SignalNotification** XJMF Signals SHALL be subscribed for by using the standard subscription mechanisms described in Section 11.5.3, “Managing Persistent Channels.” on page 1127.

11.4 XJDF and XJMF Interchange Protocol

XJDF and **XJMF** SHOULD be exchanged over a network by using http [RFC2616] or https. Controllers and Devices SHOULD implement an http server.

Note: A Controller or Device could be designed without an http server by polling but this is discouraged.

Controllers and Devices SHOULD provide insecure http without an SSL layer for better interoperability. Controllers and Devices MAY provide hot folders or other file based mechanisms for exchange of **XJDF** or **XJMF** for debugging and prototyping purposes.

Note: It is strongly discouraged to design a production workflow based on hot folders.

11.4.1 HTTP Port

XJMF Messaging does not specify a standard port.

11.4.2 HTTP Request Method

A Sender SHALL use an HTTP Post request to transmit an **XJMF** that contains **XJMF** Queries, **XJMF** Commands and **XJMF** Signals to an HTTP server.

The contents SHALL be placed in the body of the http request. See ##ref packaging below for details of **XJMF** packaging.

The Receiver SHALL place the XJMF containing XJMF Response Messages in the body of the response to the HTTP post. The Receiver SHALL package Response messages as raw XML.

The body of an HTTP response to an XJMF that contains only XJMF Signals that are not defined as reliable (*@ChannelMode!="Reliable"*) MAY be empty.

HTTPS-Based Protocol – SSL Secure XJMF has no additional requirements in addition to standard SSL [SSL3].

Note: Since Controllers and Devices will typically implement the http client interface and the http server interface, Sender and Receiver will need to provide certificates and maintain the chain of trust to verify that the certificates are valid.

11.5 XJMFXJMF XJMF Handshaking

This section describes the actions and appropriate reactions in a communication between Controllers and Devices using **XJMF**.

11.5.1 Single Query/Command Response Communication

The handshaking mechanisms for queries and commands are identical. The Sender SHALL send a **Query Message** or **Command Message** to the Receiver. The Receiver SHALL parse the **Query Message** or **Command Message** and SHALL synchronously return an appropriate **Response** to the Sender. *@refID* SHALL be set to the value of *@ID* of the message from the Sender. If the incoming Message could not be parsed, the Response SHALL be a **ResponseNotification**.

11.5.1.1 XJMF Error Handling

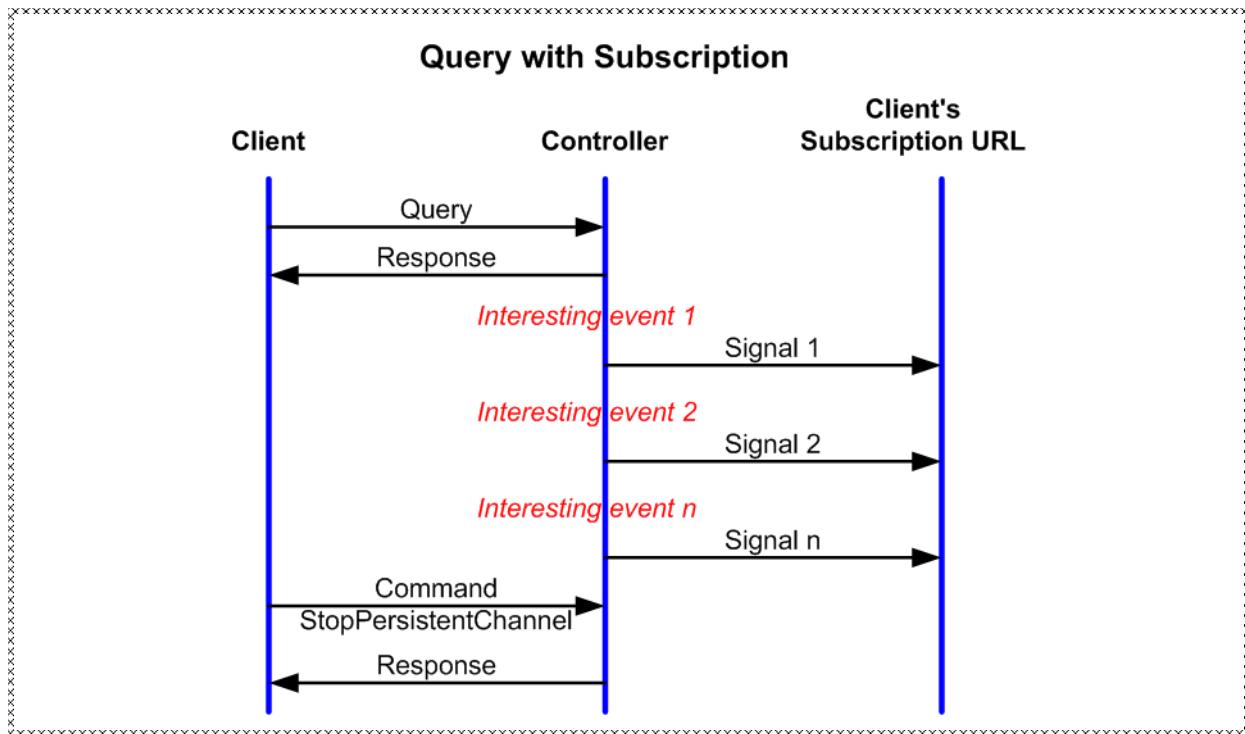
If a Command Message, Query Message, or a Signal Message is not successfully handled, a processor SHALL reply with a Response that SHALL contain a non-zero *@ReturnCode* from Appendix D, “Supported Error Codes in XJMF and Notification Elements” on page 1183 and that SHOULD contain a **Notification Element** that SHOULD provide additional details of the error.

The Response Messages contain a *@ReturnCode* Attribute. *@ReturnCode* defaults to 0, which indicates that the response is successful. In case of success and in responses to commands an informational **Notification Element** (*@Class = "Information"*) MAY be provided. In case of a warning, error or fatal error, the *@ReturnCode* is greater than 0 and indicates the kind of error committed. In this case, a **Notification Element** SHOULD be provided. Error codes are defined in Appendix D, “Supported Error Codes in XJMF and Notification Elements” on page 1183. The responding application SHOULD fill additional **Notification/Error %%Elements** that describe the details of the error.

Example 11-1: Response with Notification Element

The following example uses a **Notification Element** to describe an error:

TBD 2.x Example.

Figure 11-5: Interaction of Messages with a Subscription

11.5.2 Subscribing for Signals

Queries SHALL be subscribed by including a `Subscription` Element that defines the details of the subscription (see also Figure 5-1). The Receiver of the subscription SHALL initially send an empty Response Message to the Sender. The Receiver of the subscription SHALL send XJMF Signals whenever the conditions that were specified in the `Subscription` element are met. Such a subscribed Query that requests multiple Signals is referred to as a "persistent channel".

Note: the Sender and Receiver roles for Signals are reversed compared with the initial subscription.

If a Controller that does not support persistent channels is queried to set up a persistent channel, it SHALL answer the Query Message with a Response Message, set `@Subscribed` to "`false`", and set the `@ReturnCode` to "`111`".

Multiple Attributes of a `Subscription` Element are combined as a Boolean OR operation of these Attributes. For instance, if `@RepeatStep` and `@ObservationTarget` are both specified, Messages fulfilling either of the requirements are requested. If the `Subscription` Element contains only a URL, it is up to the emitting Controller to define when to emit Messages.

11.5.3 Managing Persistent Channels.

A Controller MAY request information about currently active subscriptions by sending a `QueryKnownSubscriptions` to a Device. A Controller SHOULD NOT send a new `Subscription` if a matching `Subscription` is already in place in the Device. If the device does not support `QueryKnownSubscriptions`, the Controller MAY create a new `Subscription`. A Device that receives a `Subscription` of the same type to the same URL SHOULD replace the existing `Subscription` with the new `Subscription`.

A Controller SHOULD remove persistent channels that are no longer evaluated by sending a `CommandStopPersistentChannel` to a Device.

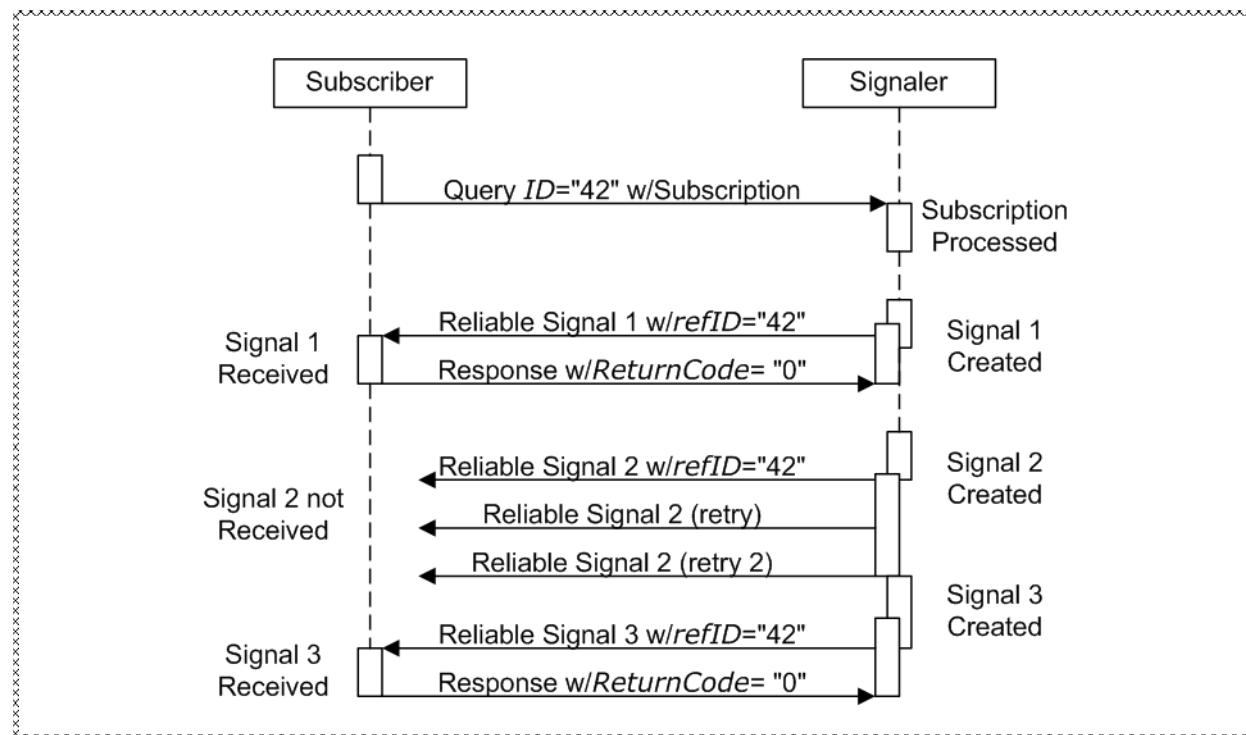
11.5.4 Signal Handshaking

JMF Signal Messages that were subscribed with `Subscription/@ChannelMode = "FireAndForget"` SHALL NOT be resent in case they were not successfully delivered to the Signal Receiver. In case of success, the Receiver SHALL send an HTTP response with an empty body. If an error occurred, the Receiver SHOULD return an error Response Message as defined in Section 11.5.1.1, “XJMF Error Handling” on page 1126.

11.5.5 Reliable Signalling

JMF Signal Messages Messages that were subscribed with `Subscription/@ChannelMode = "Reliable"` SHALL be resent in case they were not successfully delivered to the Signal Receiver. If the receiver does not respond to the reliable signal, the sender SHALL retry the reliable signal, based on the `@RetryPolicy` specified in the original `Subscription` Element. If a Response is received with a `@ReturnCode` value other than zero, then the Signal Message SHOULD be retried, unless the sender determines that resending the message is not useful in an implementation specific manner.

Figure 11-6: Example of Reliable Signaling



11.5.5.1 12.3.4.1 Sequence of Signals

Signals SHALL be sent in the order that the underlying event that triggered the Signal occurred. Thus subsequent Signals that occurred after a Signal that was in error and needs to be resent SHALL NOT be sent until either the offending Signal has been successfully resent or the offending Signal has been discarded.

11.5.6 Deleting Persistent Channels

A persistent channel is to be deleted by sending a `StopPersistentChannel` Command Message, as described in Section 5.5.8, “`StopPersistentChannel`”.

11.5.7 XJMF Bootstrapping

XJMF currently provides no mechanism for initial Device discovery. Thus the URL of an XJMF Device needs to be provided to a Controller outside of XJMF. Once the **XJMF** URL is known, a Controller SHOULD follow the steps specified in Table 11-3, “XJMF Bootstrapping steps” to initiate communication.

Table 11-3: XJMF Bootstrapping steps

Sender	Message	Remarks
Controller	QueryKnownMessages	The Controller SHOULD query for known messages and refrain from sending unknown messages, including messages specified in this section.
Device	ResponseKnownMessages	The Controller SHOULD respond with a list of known messages.
Controller	QueryKnownDevices	If the Device is a workflow controller, the Controller MAY query for additional known lower-level devices that the Device wishes to publish. The Controller SHOULD apply this bootstrapping procedure defined in this section to all lower level devices that are supplied in the ResponseKnownDevices.
Device	ResponseKnownDevices	The Device SHOULD respond with a list of known lower-level devices.
Controller	QueryKnownSubscriptions	If the Controller intends to subscribe for Signals from the Device, the Controller SHOULD query for a list of existing subscriptions. The Controller SHOULD NOT resubscribe for existing subscriptions.
Device	ResponseKnownSubscriptions	The Device SHOULD respond with a list of known subscriptions.

11.5.8 Device / Controller Selection

XJMF defines the **KnownDevices** Query Message to find Controllers and Devices. The information provided by this query can be used by a Controller to infer the appropriate routing for a Node. In a system that does not support messaging, this information SHOULD be provided outside of **XJDF**.

11.6 XJDF Packaging

An XJMF MAY be transferred with no additional packaging. Alternatively, an XJMF and its referenced assets MAY be combined into a single zip package consisting of:

- a single **XJMF** Message,
- the **XJDF** Job tickets to which it refers, and
- the digital assets to which the **XJMF** and **XJDF** Job tickets refer.

XJDF messages that do not refer to external assets SHOULD NOT be packaged as zip. XJDF MAY refer to digital assets that are not included in the zip package. Multiple XJMF messages SHALL NOT be packaged in one zip package.

11.6.1 MIME Types and File Extensions

The following MIME types and extensions SHOULD be when storing XJDF or XJMF as files or when a MIME type is required, e.g. when setting the http Content-Type header. The MIME type for **XJDF** is not yet registered with IANA <http://www.iana.org/>. The registration process is ongoing and the MIME types will be registered according to table:

||MIME Type||Extension||Usage||

application/vnd.cip4-xjdf+xml	xjdf	unpackaged XJDF
application/vnd.cip4-xjmf+xml	xjmf	unpackaged XJMF
application/vnd.cip4-xjdf+zip	zip	zip packaged XJDF
application/vnd.cip4-xjmf+zip	zip	zip packaged XJMF

Device or **XJDF**Controllers and Devices **XJMF**

11.6.2 ZIP Packaging

Zip is a de facto industry standard for packaging and compressing data. Directory structures can be encoded in a ZIP package. Details can be found at: <https://www.pkware.com/support/zip-app-note/>.

11.6.2.1 Identifying the Root XJMF.

The root XJMF SHALL be named root.xjmf and SHALL reside in the root directory of the ZIP package.

11.6.2.2 Referencing Assets within a ZIP Package.

Referenced assets that reside in the ZIP package, for instance those that are referenced with //*/@URL, SHALL be referenced as local URLs. The current URL for calculating local URLs SHALL be the root of the ZIP package, regardless of the location of the referring asset within the ZIP package.

Assets other than the root.xjmf MAY be placed in a directory tree structure within the ZIP file.

```
XJDFXJDFXJMFMIME-Version: 1.0
Content-Type: multipart/related; boundary=abcdefg0123456789
Content-Transfer-Encoding: 8bit
--abcdefg0123456789
Content-Type: application/vnd.cip4-jdf+xml
<JDF ... >
  <PreviewImage Separation = "PANTONE 128" URL="cid:123456.png" />
</JDF>
--abcdefg0123456789--

MIME-Version: 1.0
Content-Type: multipart/related; boundary=abcdefg0123456789
Content-Transfer-Encoding: 8bit
--abcdefg0123456789
Content-Type: application/vnd.cip4-jdf+xml

<JDF ... >
  <PreviewImage Separation="PANTONE 128" URL="cid:123456.png@cip4.org" />
</JDF>

--abcdefg0123456789
Content-Type: image/png
Content-Transfer-Encoding: base64
Content-ID: <123456.png@cip4.org>

BASE64DATA
BASE64DATA

--abcdefg0123456789--

MIME-Version: 1.0
Content-Type: multipart/related; boundary=unique-boundary

--unique-boundary
Content-Type: application/vnd.cip4-jmf+xml
Content-Transfer-Encoding: 8bit
...
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderId="JMFClient"
```

```

"2005-07-07T13:15:56+01:00" Version="1.4"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<Command ID="C0001" Type="SubmitQueueEntry" xsi:type="CommandSubmitQueueEntry">
  <QueueSubmissionParams Hold="true" URL="cid:JDF1@hostname.com"/>
</Command>
</JMF>
--unique-boundary
Content-Type: application/vnd.cip4-jdf+xml
Content-Transfer-Encoding: 8bit
Content-ID: <JDF1@hostname.com>
Content-Disposition: attachment; filename="Ticket01.jdf";
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" Activation="Active" ID="JDF_c"
      JobID="Geef62b72-0f6e-4195-a412-aaa3123d200b" Status="Waiting" Type="Product"
      Version="1.4" JobPartID="345">
  <ResourceLinkPool>
    <ComponentLink Usage="Output" rRef="ID125"/>
  </ResourceLinkPool>
  <ResourcePool>
    <RunList Class="Parameter" DocCopies="1" FirstPage="0" ID="RunList4"
            IsPage="true" NDoc="1" PageCopies="1" Status="Available">
      <LayoutElement ElementType="Document" HasBleeds="false" ID="LayoutElement_1"
                     IgnorePDCopies="true" IgnorePDLImposition="true" IsPrintable="true">
        <FileSpec AppOS="Windows" Compression="None" Disposition="Retain"
                  ID="FileSpec_9" URL="cid:Asset01@hostname.com"
                  UserFileName="Christmas Cards"/>
      </LayoutElement>
    </RunList>
    <Component ID="ID125" Class="Quantity" Status="Unavailable"
               ComponentType="Sheet" />
  </ResourcePool>
  <JDF ID="JDF-3" Status="Waiting" Type="DigitalPrinting" JobPartID="400">
    <ResourceLinkPool>
      <DigitalPrintingParamsLink Usage="Input" rRef="ID123"/>
      <RunListLink Usage="Input" rRef="RunList4"/>
      <ComponentLink Usage="Output" rRef="ID125"/>
    </ResourceLinkPool>
    <ResourcePool>
      <DigitalPrintingParams ID="ID123" Class="Parameter" Status="Available" />
    </ResourcePool>
  </JDF>
</JDF>

--unique-boundary
Content-type: application/pdf
Content-ID: <Asset01@hostname.com>
Content-Transfer-Encoding: binary
Content-Disposition: attachment; filename="Pages 1.pdf";

The pdf goes in here.
--unique-boundary--

```

11.7 XJob Modification

While jobs are waiting for execution in a queue or even during execution of a job, circumstances may arise that require modifications to that job. XJDF enables modifications to jobs using the `ModifyQueueEntry` and `ResubmitQueueEntry` messages.

Note: Although the XJDF mechanisms for modifying jobs are fairly simple, the underlying physical changes may make modifications difficult or even impossible. The actual implementation of changes is always device dependent, and Controllers SHOULD always expect modification requests to fail and process failure appropriately.

11.7.1 Rescheduling with ModifyQueueEntry

`ModifyQueueEntry` is designed to allow rescheduling of jobs without changing any job's. Typical use cases for `ModifyQueueEntry` are:

- Reordering the sequence of execution to optimize setup times by running similar jobs in sequence;
- Suspending a running job so that a rush job can be processed before the current job is completed;

11.7.2 Modifying Jobs

`ResubmitQueueEntry` is designed to modify the `s` of the underlying sets of XJDF nodes of a job. 4 types of job modification are supported. These are differentiated by `ResubmissionParams/@UpdateMethod` and the related `XJDF/@JobPartID`

Table 11-4: Modifying Job Parameters

<code>ResubmissionParams/@UpdateMethod</code>	<code>XJDF/@JobPartID</code>	Description
Complete	-	If <code>XJDF/@JobPartID</code> is omitted, the job parameters of all XJDFs that belong to the queue entry SHALL be completely overwritten with new information.
Complete	known	The job parameters relating to <code>@JobPartID</code> SHALL be completely overwritten with new information.
Incremental	known	The job parameters that relate to the existing XJDF are overwritten by the data that is explicitly supplied in the referenced XJDF.
Incremental	new	A new process step is requested. Details SHALL be supplied in the referenced XJDF.
Delete	known	The job parameters that are explicitly supplied in the referenced XJDF SHALL be removed. If no XJDF is provided, the entire process step described by <code>@JobPartID</code> SHALL be removed.

Typical use cases for `ResubmitQueueEntry` are:

- Change the number of copies requested;
- Change the number or details of physical inks required for printing;
- Change content data such as number of pages or page size;
- Change the details of the physical substrate to print on;
- Change binding or other finishing options;
- Select a different device with differing properties, e.g. sheet size, to optimize utilization of multiple devices.

11.8 Use of XML Schema for Capability Descriptions

Individual devices will never implement the entire XJDF specification. Meaningful communication between a Controller and a Device is only possible if the Controller is aware of the limitations of the Device.

XJDF does not provide a proprietary method to define Device capabilities. Since XJDF is an XML dialect, standard XML tools such as XML schema [XMLSchema] SHOULD be used to declare the supported features of a device.

CIP4 provides schema for the entire XJDF specification and reduced schema for ICS documents. Vendors are encouraged to provide XML schemas that define the supported XJDF features of their Devices.

Device schema for XJDF SHALL use the XJDF namespace for standard XJDF features.

Appendix A Encoding

This appendix lists a number of commonly used **XJDF** data types and structures and their XML encoding. Data types are simple data entities such as strings, numbers (as doubles) and dates. They have a very straightforward string representation and are used as XML Attribute Values. Data structures, on the other hand, describe more complex structures that are built from the defined data types, such as colors.

A.1 Notes About Encoding

All of the **XJDF** types are derived from XML Schema types, either by extension, use of lists or by restriction. Each type will refer back, either directly or indirectly, to such a type and reference ought to be made to “XML Schema Part 2 – Datatypes” [XMLSchema].

A.1.1 List, Range and Range List Data Types

Some data types are derived from a base type that represents a single value. Such data types include a list, a range and a range list. For a data type *X*, the name of such data types are *XList*, *XRange* and *XRangeList*, respectively. Each data type represents a set of values of the base data type. A range consists of a pair of (possibly) compound values, where each (compound) value is specified by one or more values (possibly a number), each separated by whitespace character. For example, a rectangle is a compound value that is specified by 4 numbers. A list is an enumerated set of values, which is expressed as a list of space separated values. A range is a continuous inclusive range of values, which is expressed as a pair of values separated by a whitespace character. A range list is expressed as a list of space separated ranges.

A.1.2 Whitespace

The addition of whitespace characters for single types is NOT RECOMMENDED. Items in a list of values are separated by whitespace.

A.1.3 Infinity Limits

Several types require the ability to set an unbounded range, or to select a single terminating value (e.g., Integer or date ranges). These types have been extended with the tokens “*-INF*” or “*INF*” to indicate the maximum negative and positive limits of the values in question, details are shown where appropriate for each value.

A.2 Simple Types — Attribute Values

A.2.1 boolean

Has the value space for supporting the mathematical concept of binary-valued logic:

Encoding

boolean Attributes are encoded as either of the string values “*true*” or “*false*”. The XML Schema data type boolean values of “*1*” or “*0*” are not permitted.

Example A-1: boolean

```
<Example Enable="true"/>
```

A.2.2 CMYKColor

XML Attributes of type CMYKColor are used to specify CMYK colors.

Encoding

CMYKColor Attributes are primitive data types and are encoded as a string of four numbers (as doubles) in the range of [0...1.0] separated by whitespace. A value of 0.0 specifies no ink and a value of 1.0 specifies full ink. The sequence of colors is “C M Y K”.

Example A-2: CMYKColor

```
<Color cmyk = "0.3 0.6 0.8 0.1"/><!--brick red-->
```

A.2.3 date

A calendar date, it represents a time period that starts at midnight on a specified day and lasts for 24 hours. Based on [ISO8601:2004].

Encoding

It is represented identically to the XML Schema type: *date*

Example A-3: date

```
<Example StartDate="1999-05-31"/>
```

A.2.4 dateTime

Represents a specific instant of time. It SHALL be a Coordinated Universal Time (UTC) or the time zone SHALL be indicated by the offset to UTC. In other words, the time SHALL be unique in all time zones around the world. It also allows infinity limits to allow for explicit ‘don’t care’ values (i.e., it SHALL be finished before ‘anytime’).

Encoding

It is represented as a union of the XML Schema type: *dateTime* and the infinity value tokens *INF* and *-INF*.

Note that [ISO8601:2004] allows a wider range of time zone specifications than XML. *dateTime* SHALL adhere to the stricter limitations defined in [XMLSchema]. For instance the colon ‘:’ in the time zone field SHALL be present when writing time zones in the format “*hh:mm*”.

Example A-4: dateTime

```
<Example Start="1999-05-31T18:20:00Z"/>
<Example Start="1999-05-31T13:20:00-05:00"/>
```

A.2.5 DateTimeRange

XML Attributes of type *DateTimeRange* are used to describe a range of points in time. More specifically, it describes a time span that has an absolute start and end. Unbounded ranges can use the infinity value tokens *INF* and *-INF*

Encoding

A *DateTimeRange* is represented by two *dateTime* or infinity tokens separated by the whitespace.

Example A-5: DateTimeRange

```
<XXX range="1999-05-31T18:20:00Z 1999-05-31T18:20:00Z"/>
<XXX range="1999-05-31T18:20:00Z INF"/>
<XXX range="-INF 1999-05-31T18:20:00Z"/>
```

A.2.6 DateTimeRangeList

XML Attributes of type *DateTimeRangeList* are used to describe a list of ranges of points in time. More specifically, it describes a list of time spans, which each have a relative start and end.

Encoding

A *DateTimeRangeList* is represented by sequence of either *DateTimeRange* values (See 1.5), separated by whitespace or *dateTime* values.

Example A-6: DateTimeRangeList

```
<XXX RangeList=
      "1999-05-31T18:20:00Z 1999-05-31T18:20:00Z 1999-05-31T13:20:00-05:00 INF"/>
```

A.2.7 double

double Corresponds to IEEE double-precision 64-bit floating point type. It includes the infinity limit tokens *INF* and *-INF*, but does not allow the not a number token *Nan*.

Encoding

It is represented similarly to the XML Schema type: *double*. However string value *Nan*, is not permitted.

Example A-7: double

```
<Example NegativePi="-3.14" />
```

A.2.8 DoubleList

XML Attributes of type DoubleList are used to describe a variable length list of numbers (as doubles). This type is used as the base for other XJDF types that use a fixed length list of number (e.g., CMYKColor which is restricted to four number in the list).

Encoding

A DoubleList is encoded as a string of whitespace-separated double values as defined in Section A.2.7, “double”.

Example A-8: DoubleList

```
<XXX list="3.14 1 .6" />
```

space

A.2.9 duration

Represents a duration of time. Based on [ISO8601:2004]. The single infinity limit token *INF* is permitted.

Encoding

It is represented as a union of the XML Schema type: *duration* and the string value *"INF"*

Note that [XMLSchema] explicitly allows negative durations. Thus a value of -PT15M is valid and describes a negative duration of 15 minutes in the past.

Example A-9: duration

```
<Example Duration= "P1Y2M3DT10H30M" />
```

A.2.10 DurationRange

XML Attributes of type DurationRange are used to describe a range of time durations. More specifically, it describes a time span that has a relative start and end.

Encoding

A DurationRange is represented by two duration values, separated by the space character and OPTIONAL additional whitespace.

Example A-10: DurationRange

```
<XXX range="P1Y2M3DT10H30M P1Y2M3DT10H35M" />
<XXX range="P1Y2M3DT10H30M INF" />
```

A.2.11 DurationRangeList

XML Attributes of type DurationRangeList are used to describe a list of ranges of time durations. More specifically, it describes a list of time spans that have a relative start and end.

Encoding:

A DurationRangeList is represented by sequence of DurationRange values and durations, separated by whitespace.

Example A-11: DurationRangeList

```
<XXX RangeList="P1Y2M3DT10H30M P1Y2M3DT10H35M P1Y3M2DT10H30M P1Y3M2DT10H30M "/>
```

A.2.12 gYearMonth

Represents a specific Gregorian month in a specific Gregorian year. Based on [ISO8601:2004].

Encoding

It is represented identically to the XML Schema type: *gYearMonth*

Example A-12: gYearMonth

```
<Example Month="2002-11"/>
```

A.2.13 hexBinary

Represents arbitrary hex encoded binary data.

Encoding

It is represented identically to the XML Schema type: *hexBinary*

Example A-13: hexBinary

```
<Example Hex="0A1C"/>
```

A.2.14 ID

Represents the *@ID* Attribute from [XMLSchema]. It represents a name or string that contains no space characters and starts with a letter, or ‘_’. Each ID value SHALL be unique within an XJDF document and thus uniquely identify the elements that bear them.

Note that the *@ID* Attribute definition in [XMLSchema] is more restrictive than the *@ID* Attribute definition in [XML]. [XMLSchema] explicitly forbids the use of ‘.’ in ID.

Encoding

It is represented identically to the XML Schema type: *ID*

Example A-14: ID

```
<Example ID="R-16"/>
```

A.2.15 IDREF

IDREF Represents the IDREF Attribute from [XMLSchema]. For a valid XML-document, an element with the ID value specified in IDREF SHALL be present in the scope of the document.

Encoding

It is represented identically to the XML Schema type: *IDREF*

Example A-15: IDREF

```
<Example IDREF="R-16"/>
```

A.2.16 IDREFS

IDREFS Represents the IDREFS Attribute from [XMLSchema]. More specifically, this is a whitespace-separated list of IDREF values.

Encoding

It is represented identically to the XML Schema type: *IDREFS*

Example A-16: IDREFS

```
<Example IDREFS="R-12 R-16"/>
```

A.2.17 integer

Represents numerical integer values with tokens for representing infinity limits.

Implementation note: Except where explicitly noted otherwise, integers are not expected to exceed a value that can be represented as signed 32 bits.

Encoding

It is represented as a union of the XML Schema type: *integer* and the infinity value tokens *INF* and *-INF*

Example A-17: integer

```
<Example Copies="36"/>
```

A.2.18 IntegerList

XML Attributes of type IntegerList are used to describe a variable length list of integer values.

Encoding

An IntegerList is encoded as a string of integers separated by whitespace.

Example A-18: IntegerList

```
<XXX list="-INF 0 1 2 3 4 INF 1 3 0"/>
```

A.2.19 IntegerRange

XML Attributes of type IntegerRange are used to describe a range of integers. In some cases, ranges are defined for an unknown number of objects. In these cases, a negative value denotes a number counted from the end. For example, -1 is the last object, -2 the second to last and so on. IntegerRanges that follow this convention are marked in the respective Attribute descriptions.

If the first element of an IntegerRange specifies an element that is behind the second element, the Range specifies a list of integers in reverse order, counting backwards. For example "6 4" = 6 5 4 and "-1 0" = "last... 2 1 0".

Encoding

An IntegerRange is represented by two integers, separated by a space character and OPTIONAL additional whitespace.

Example A-19: IntegerRange

```
<XXX range="-3 -5"/>
<XXX range="INF -5"/>
```

A.2.20 IntegerRangeList

XML Attributes of type IntegerRangeList are used to describe a list of IntegerRanges.

Encoding

An IntegerRangeList is represented by a sequence of IntegerRanges, separated by whitespace.

Example A-20: IntegerRangeList

```
<XXX list="-1 -6 3 5 7 7 9 128 131 131"/>
```

A.2.21 LabColor

XML Attributes of type LabColor are used to specify absolute Lab colors. The Lab values are normalized to a Light of D50 and an angle of 2 degrees as specified in [CIE 15:2004] and [ISO13655:1996].

This corresponds to a white point of X = 0.9642, Y = 1.0000 and Z = 0.8249 in CIEXYZ color space. The value of L is restricted to a range of [0..100]; a and b are unbounded.

Encoding

LabColors are primitive data types and are encoded as a list of three numbers (as doubles) separated by whitespace in the sequence: “L a b”

Example A-21: LabColor

```
<Color Lab="51.9 12.6 -18.9"/>
```

A.2.22 language

Represents a natural language defined in [RFC1766].

Encoding

It is represented identically to the XML Schema type: *language*

Example A-22: language

```
<Example Language="de"/> <!-- German -->
<Example Language="de-CH"/> <!-- Swiss German -->
<Example Language="en"/> <!-- English -->
<Example Language="en-GB"/> <!-- British English -->
```

A.2.23 languages

XML Attributes of type languages are used to describe a variable length list of language values.

Encoding

A languages value is encoded as a string of languages, each language separated by whitespace.

Example A-23: languages

```
<Example Languages ="de-CH de en-GB en"/>
```

A.2.24 matrix

Coordinate transformation matrices are widely used throughout the whole printing Process, especially in **Layout** Resources. They represent two dimensional transformations as defined by [PS] and [PDF1.6]. For more information, refer to the respective reference manuals, and look for “Coordinate Systems and Transformations.” The “identity matrix”, which is “1 0 0 1 0 0”, is often used as a default throughout this specification. When another matrix is factored against a matrix with the identity matrix value, the result is that the original matrix remains unchanged.

Encoding

Coordinate transformation matrices are primitive data types and are encoded as a list of six numbers (as doubles), separated by whitespace: “a b c d Tx Ty”. The variables *Tx* and *Ty* describe distances and are defined in points.

Example A-24: matrix

```
<ContentObject CTM="1 0 0 1 3.14 21631.3" />
```

A.2.25 NameRange

XML Attributes of type NameRange are used to describe a range of NMTOKEN data that are acquired from a list of named elements, such as named pages in a PDL file. It depends on the ordering of the targeted list, which names are assumed to be included in the NameRange. The following two possibilities exist:

- 1 There is no explicit ordering. In this case, case sensitive alphabetical ordering [Unicode5.0] is implied. This behavior is the default unless called out explicitly in the specification.
- 2 There is explicit ordering, such as in a list of named pages in a **RunList**. In this case, the ordering of the **RunList** defines the order and all pages between the end pages are included in the NameRange.

Encoding

A NameRange typed Attribute is represented by two NMTOKEN values separated by a space character and OPTIONAL additional whitespace.

Example A-25: NameRange

```
<XXX NameRange="Jack Jill"/>
```

A.2.26 NameRangeList

XML Attributes of type NameRangeList are used to describe a list of NameRanges.

Encoding

A NameRangeList is represented by a sequence of NameRanges, separated by whitespace.

Example A-26: NameRangeList

```
<XXX list="A A brian fred x x z z"/>
```

A.2.27 NMTOKEN

Represents the NMTOKEN Attribute type from [XML]. It represents a name or string that contains no space characters.

Encoding

It is represented identically to the XML Schema type: *NMTOKEN*

Example A-27: NMTOKEN

```
<Example Alias="ABC_6"/>
```

A.2.28 NMTOKE NS

NMTOKE NS Represents the NMTOKE NS Attribute type from [XML]. More specifically, this is a whitespace-separated list of NMTOKEN values.

Encoding

It is represented identically to the XML Schema type: *NMTOKE NS*

Example A-28: NMTOKE NS

```
<Example AliasList="ABC_6 ABCD_3 DEGF"/>
```

A.2.29 PDFPath

XML Attributes of type PDFPath are used in XJDF for describing parameters such as trap zones and clip paths. In PJTF, PDFPaths are encoded as a series of **moveto-lineto** operations. XJDF has a different encoding, which is able to describe more complex paths, such as Bezier curves. The non-zero winding rule is used to fill closed paths.

Encoding

PDFPaths are encoded by restricting an XML *string* Attribute formatted with PDF path operators. This allows for easy adoption in PS and PDF workflows. PDF operators are limited to those described in “Path Construction Operators” in [PDF1.6].

Example A-29: PDFPath

```
<ElementWithPath path="0 0 m 10 10 l 20 20 l" />
```

A.2.30 rectangle

XML Attributes of type rectangle are used to describe rectangular locations on the page, Sheet or other printable surface. A rectangle is represented as an array of four numbers — llx lly urx ury — specifying the lower-left x, lower-left y, upper-right x and upper-right y coordinates of the rectangle, in that order. This is equivalent to the ordering: Left Bottom Right Top. All numbers are defined in points.

Encoding

To maintain compatibility with PJTF, rectangles are primitive data types and are encoded as a string of four *numbers*, separated by whitespace: "llx lly urx ury" or "l b r t".

Example A-30: rectangle

```
<ContentObject ClipBox="0 0 3.14 21631.3" />
```

Implementation Remark

Since all numbers are real numbers, any comparison of boxes SHOULD take into account certain rounding errors. For example, different XYPair values MAY be considered equal when all numbers are the same within a range of 1 point.

A.2.31 RectangleRange

XML Attributes of type RectangleRange are used to describe a range of rectangles.

Encoding

A RectangleRange is represented by one or two Rectangles, separated by a space character and OPTIONAL additional whitespace.

Example A-31: RectangleRange

```
<XXX range="1 2 3 4 5 6 7 8" />
<XXX range="-INF -INF 3 4 0 1 INF INF" />
```

A.2.32 regExp

Represents a regular expression as defined in [XMLSchema].

Encoding

It is represented identically to the XML Schema type: *normalizeString*

Example A-32: regExp

```
<Example expression="Foo({1|2}*)" />
```

A.2.33 shape

XML Attributes of type shape are used to describe a three dimensional box.

Encoding

A shape is represented as an array of three (positive or zero) *numbers* — x y z — specifying the Width x, height y and depth z coordinates of the shape, in that order.

Example A-33: shape

```
<XXX Dimensions="10 20 40" />
```

A.2.34 ShapeRange

XML Attributes of type ShapeRange are used to describe a range of shapes (three dimensional boxes). The range " $x_1 \ y_1 \ z_1 \ x_2 \ y_2 \ z_2$ " describes the area $x_1 \leq x \leq x_2$ and $y_1 \leq y \leq y_2$ and $z_1 \leq z \leq z_2$. Thus the shape "2 3 4" is within "1 2 1 3 4 4".

Note that this implies that all three values of the second entry SHALL be \geq the corresponding values of the first entry. The following example is therefore invalid: "1 2 1 0 4 4".

Encoding

Example A-34: A ShapeRange is represented by two shapes, separated by a space character and OPTIONAL additional whitespace. Note: **ShapeRange**

```
<XXX Shaperange="1 2 3 4 5 6"/>
<XXX Shaperange="1 2 3 4 INF 6"/>
```

[] [] []

A.2.35 sRGBColor

XML Attributes of type sRGBColors are used to specify sRGB colors.

Encoding

sRGBColors are primitive data types and are encoded as a string of three numbers in the range of [0...1.0] separated by whitespace. A value of 0 specifies no intensity (black) and a value of 1 specifies full intensity. The sequence is defined as: "r g b"

Example A-35: sRGBColor

```
<Color sRGB="0.3 0.6 0.8" />
```

A.2.36 string

Represents character strings in XML.

Encoding

It is represented identically to the XML Schema type: *normalisedString NB*. This means that tabs, linefeeds and so on are not valid characters.

Example A-36: string

```
<Example Name="Test With Space"/>
```

A.2.37 TransferFunction

XML Attributes of type TransferFunction are functions that have a one-dimensional input and output. In XJDF, they are encoded as a simple kind of sampled functions and used to describe transfer curves of image transfer Processes from one medium to the next (e.g., film to plate, or plate to press).

A transfer curve consists of a series of XY pairs where each pair consist of the stimuli (X) and the resulting value (Y). To calculate the result of a certain stimuli, the following algorithms SHALL be applied:

- 1 If $x \leq$ first stimuli, then the result is the y value of the first xy pair.
- 2 If $x \geq$ the last stimuli, then the result is the y value of the last xy pair.
- 3 Search the interval in which x is located.
- 4 Return the linear interpolated value of x within that interval.

Encoding

A TransferCurve is encoded as a string of space-separated *numbers* (as doubles). The numbers are the XY pairs that build up the transfer curve. Note that the end points of a TransferFunction SHALL be explicitly specified and are NOT defaulted to "0 0" or "1 1".

Example A-37: TransferFunction

```
<someElementWithTransferCurve someCurve="0 0 .1 .2 .5 .6 .8 .9 1 1"/>
```

A.2.38 URI

Short for URI-reference. Represents a Uniform Resource Identifier (URI) Reference as defined in [RFC3986]. The URI data typed is represented as an Internationalized Resource Identifier (IRI) as defined in [RFC3987].

Encoding

A URI is represented identically to the XML Schema type: *anyURI*.

Example A-38: URI

```
<Example URI="http://www.w3.org/1999/XMLSchema"/>
```

A.2.39 URL

Short for URL-reference. Represents a Uniform Resource Locator (URL) Reference as defined in [RFC3986]. The URL data typed is represented as an Internationalized Resource Identifier (IRI) as defined in [RFC3987].

Encoding

A URL is represented identically to the XML Schema type: *anyURI*.

Some characters in a URL SHALL be escaped and all characters MAY be escaped by encoding their UTF-8 representation into a '%' followed by the double digit hex representation of the character. The list of characters that SHALL be encoded is dependent on the URL scheme. Non-escaped characters SHALL be encoded in the encoding of the containing XJDF document.

Example A-39: URL

A UNC path to be displayed as a URL:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Example URL="\\\\\\myHost\\\\a\\c äöü%.pdf"/>
```

Example A-40: URL: UTF-8

The UNC path encoded as an IRL with internationalized characters in UTF-8:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<Example URL="file://myHost/a/c%20ä%25. pdf"/>
```

Example A-41: URL: Windows Locale 1252

The same UNC path encoded as an IRL with internationalized characters in UTF-8 viewed in a Windows locale 1252:

```
<Example URL="file://myHost/a/c%20ä%25.pdf"/>
```

Example A-42: URL: Escaped Characters

The same UNC path encoded as an IRL with internationalized characters escaped:

```
<Example URL="file://myHost/a/c%20%c3%a4%c3%b6%c3%bc%25.pdf"/>
```

A.2.40 XPath

Represents an XPath expression. [XPath]

Encoding

It is represented identically to the XML Schema type: *token*

Example A-43: XPath

```
<Example xpath= "JDF/AuditPool/Created/@TimeStamp" />
```

A.2.41 XYPair

XML Attributes of type XYPair are used to describe sizes like *@Dimensions* and *@StartPosition*. They can also be used to describe positions on a page. All numbers that describe lengths are defined in points.

Encoding

XYPair Attributes are primitive data types and are encoded as a string of two *numbers*, separated by whitespace: “x y”

Example A-44: XYPair

```
<Media Dimension ="1984 2834"/>
```

Implementation Remark

Since all numbers are real numbers, comparison of XYPair values SHOULD take into account certain rounding errors. For example, different XYPair values MAY be considered equal when all numbers are the same within a range of 1 point.

A.2.42 XYPairRange

XML Attributes of type XYPairRange are used to describe a range of XYPair values. The range describes the area $x_1 \leq x \leq x_2$ and $y_1 \leq y \leq y_2$. Thus the XYPair “2 3” is within “1 2 3 4”. Note that this implies that both values of the second entry SHALL be \geq the corresponding values of the first entry. The following example is therefore invalid: “1 2 0 4”.

Encoding

An XYPairRange is represented by two XYPair values, separated by a space character and OPTIONAL additional whitespace. Note:

Example A-45: XYPairRange

```
<XXX XYrange="1 2 3 4"/>
<XXX XYrange="-INF 2 3 INF"/>
```

[] [] [] []]

A.3 Enumerations and Lists

A.3.1 enumeration

Represents a closed set of values.

Encoding

It is represented by an enumerated list of values derived from the XML Schema type: *NMTOKEN*

Example A-46: enumeration

```
<Example Rotate="Rotate90"/>
```

A.3.2 enumerations

Represents a list of values taken from a closed set. Values MAY be repeated within the list. If there are any implications to the order of the values this will be detailed in the appropriate items description, otherwise none is implied.

Encoding

It is represented by a whitespace-separated list of enumeration values derived from the XML Schema type: *NMTOKEN*

Example A-47: enumerations

```
<Example AcknowledgeType="Applied Completed"/>
```

A.3.3 Defined XJDF enumeration Data Types

This section is a list of defined enumeration data types. These types are to be used wherever possible for enumerated values and lists of values.

A.3.3.1 Anchor

Attributes with a data type of Anchor describe the 9 anchor points of a rectangle.

Table A-1: Anchor Enumeration Values

Enumeration Value	Comment
<i>TopLeft</i>	
<i>TopCenter</i>	
<i>TopRight</i>	
<i>CenterLeft</i>	
<i>Center</i>	
<i>CenterRight</i>	
<i>BottomLeft</i>	
<i>BottomCenter</i>	
<i>BottomRight</i>	

A.3.3.2 JDFJMFileVersion

Describes the schema version of an XJDF or XJMF instance.

Table A-2: JDFJMFileVersion Enumeration Values

Enumeration Value	Comment
1.1	JDF 1.1
1.2	JDF 1.2
1.3	JDF 1.3
1.4	JDF 1.4
1.5	JDF 1.5
2.0	XJDF 2.0

A.3.3.3 NamedColor

Machine readable definition of a color. For a list of allowed values see:

<http://www.w3.org/TR/html4/types.html#h-6.5>

A.3.3.4 Orientation

Orientation of a Resource. For details see Table 2-4, “Matrices and Orientation values for describing the orientation of a Component” on page 39.

Table A-3: Orientation Enumeration Values

Enumeration Value	Comment
<i>Rotate0</i>	
<i>Rotate90</i>	
<i>Rotate180</i>	
<i>Rotate270</i>	
<i>Flip0</i>	
<i>Flip90</i>	
<i>Flip180</i>	
<i>Flip270</i>	

A.3.3.5 Sides

Sides defines the sides of the media or product that SHALL be imaged.

Table A-4: Sides Enumeration Values

Enumeration Value	Comment
<i>OneSided</i>	Page contents SHALL be imposed on the front side.
<i>OneSidedBack</i>	Page contents SHALL be imposed on the back side.
<i>TwoSidedHeadToFoot</i>	Page contents SHALL be imposed on the front and back sides of media Sheets so that the head (top) of the front backs up to the foot (bottom) of the back.
<i>TwoSidedHeadToHead</i>	Page contents SHALL be imposed on the front and back sides so that the head (top) of page contents back up to each other.

A.3.3.6 Scope

Attributes of type define the availability of features or resources in a device.

Table A-5: Scope Enumeration Values

Enumeration Values	Comment
<i>Allowed</i>	The feature is potentially available but currently not available without operator intervention.
<i>Job</i>	The feature is currently available within the scope of a job.
<i>Present</i>	The feature is currently available without operator intervention.

A.3.3.7 WorkStyle

Table A-6: WorkStyle Enumeration Values (Sheet 1 of 2)

Enumeration Value	Comment
<i>Simplex</i>	No turning.

Table A-6: WorkStyle Enumeration Values (Sheet 2 of 2)

Enumeration Value	Comment
<i>Perfecting</i>	Many Sheet-Fed printing presses have perfecting cylinder(s) built in. The leading edge of the print Sheet changes as the Sheet is turned by the perfecting cylinder, but the side lays remain unaltered. In this regard, this @ <i>WorkStyle</i> is similar to " <i>WorkAndTumble</i> ", but " <i>Perfecting</i> " is an in-line operation during the press run. Therefore, an additional plate (set) is needed during this press run.
<i>WorkAndBack</i>	This @ <i>WorkStyle</i> describes the printing on both sides of the substrate with a different plate (set) in the second run. After the first run the side lays are altered but the front lays stay as they were. Lays can be turned by hand or using a pile reverser. Two-plate sets are necessary for " <i>WorkAndBack</i> ".
<i>WorkAndTurn</i>	<i>"WorkAndTurn"</i> refers to the turning of the first-run Sheet for subsequent perfecting. The front lays remain unchanged but the side lays SHALL be altered. The alteration can be made by hand or using a pile turner. Turning happens after the first press run and the plate (set) is used again in the second press run, imaging the other Sheet surface.
<i>WorkAndTumble</i>	The " <i>WorkAndTumble</i> " method is also used for perfecting. The leading edge of the print Sheet changes as the Sheet is turned, but the side lays remain unaltered. Tumbling happens after the first press run and the plate (set) is used again in the second press run, imaging the other Sheet surface.
<i>WorkAndTwist</i>	Done between two press runs. The palette is twisted 180 degree before the second run is performed so that the front lay and the side lay both change. The surface to be imaged is the same at both runs. Each run prints only part of the surface. The plate (set) stay in the machine. This @ <i>WorkStyle</i> is used for saving plate or film material. It is no longer a common @ <i>WorkStyle</i> .

A.3.4 XYRelation

XML Attributes of type XYRelation define the relationship between two ordered numbers.

Table A-7: XYRelation Enumeration Values

Enumeration Value	Comment
<i>gt</i>	X > Y
<i>ge</i>	X >= Y
<i>eq</i>	X = Y
<i>le</i>	X <= Y
<i>lt</i>	X < Y
<i>ne</i>	X != Y

A.4 XJDF File Formats

This section describes the specific file formats used by **XJDF**. **XJDF** uses TIFF and JPEG file formats, as well as the PNG image file format. The following sections explain in what ways PNG is used in **XJDF**.

A.4.1 PNG Image Format

XJDF uses the PNG images for representing preview images. CIP3 defined two formats: composite CMYK and separated. With PNG, only the separated format is supported for color spaces other than RGB. The composite CMYK or spot color representations SHALL be represented as separated CMYK or spot colors. Thus, preview images are stored as separate PNG images and **XJDF** links them together. Viewable images and thumbnails can be represented as composite RGB PNG images.

References: <http://www.w3.org/Graphics/png>.

Appendix B Schema

XML Schema for **XJDF** (and **XJMF**) will be published on: <http://www.CIP4.org> The XML Schema is not sufficient to completely validate an **XJDF** Job. For example, Partitioned Resources or Process Node types as defined in **XJDF** cannot be validated by XML Schema processors. In other words, the structure of some elements depends on the context of usage which cannot be completely described by XML Schema. Thus, the XML Schema for **XJDF** will be structured in a way that it enables a pre-validation of valid **XJDF**-candidates but does not preclude all syntactically invalid files to be validated.

B.1 Using xsi:type

XML Schema permits that multiple type definitions be derived from a base type. Wherever the schema has define an element of that base type, it is possible for the document to indicate to a validator the particular derived type that it has used. This it does by using the `@xsi:type` Attribute with a value of the name of the type, where the "`xsi`" tag is associated with the Schema Instance namespace that has to be declared in the document.

Note: Use of "`xsi`" as the tag is normal practice.

Note: The selected type is namespace qualified (which permits extensions)

B.1.1 Using xsi:type with XJDF Nodes

When used with **XJDF** Nodes then all Processes defined in Section 6 are supported. Furthermore the value to be used is identical to the Process type, thus an **XJDF** Node that has a `@Type` of "`DigitalPrinting`" can inform validators to use the schema definition for **DigitalPrinting** Nodes by also setting `@xsi:type` to "`DigitalPrinting`".

Some **XJDF** Nodes are general in their nature and do not have a restricted definition (i.e., Product Intent Nodes, Combined Process Nodes and so on). General definitions with the appropriate name are provided to enable consistent use of `@xsi:type`.

The **XJDF** Schema defines types for JDF Process Nodes and **XJMF** Messages. It is RECOMMENDED that these types are used with `@xsi:type`.

Example B-1: JDF Nodes: xsi:type

TBD 2.x Example.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="BackCover"
      Status="InProgress"
      Type="DigitalPrinting" Version="1.4" JobPartID="345"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.CIP4.org/Schema/JDFSchema_1_1"
      xsi:type="DigitalPrinting">
  <ResourceLinkPool>
    <DigitalPrintingParamsLink Usage="Input" rRef="ID123"/>
    <RunListLink Usage="Input" rRef="ID124"/>
    <ComponentLink Usage="Output" rRef="ID125"/>
  </ResourceLinkPool>
  <ResourcePool>
    <DigitalPrintingParams ID="ID123" Class="Parameter" Status="Available" />
    <RunList ID="ID124" Class="Parameter" Status="Available" />
    <Component ID="ID125" Class="Quantity" Status="Unavailable"
               ComponentType="Sheet" />
  </ResourcePool>
</JDF>
```

Example B-2: XJDF Nodes: xsi:type (not in Default Namespace)

If the **XJDF** is not in the default namespace then the type name needs to be altered accordingly:

TBD 2.x Example.

```
<jdf:JDF xmlns:jdf="http://www.CIP4.org/JDFSchema_1_1" ID="BackCover"
```

```

    Status="InProgress"
    Type="DigitalPrinting" Version="1.4" JobPartID="345"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="jdf:DigitalPrinting">
<jdf:ResourceLinkPool>
    <jdf:DigitalPrintingParamsLink Usage="Input" rRef="ID123"/>
    <jdf:RunListLink Usage="Input" rRef="ID124"/>
    <jdf:ComponentLink Usage="Output" rRef="ID125"/>
</jdf:ResourceLinkPool>
<jdf:ResourcePool>
    <jdf:DigitalPrintingParams ID="ID123" Class="Parameter"
        Status="Available" />
    <jdf:RunList ID="ID124" Class="Parameter" Status="Available" />
    <jdf:Component ID="ID125" Class="Quantity" Status="Unavailable"
        ComponentType="Sheet" />
</jdf:ResourcePool>
</jdf:JDF>

```

B.1.2 Using xsi:type with XJMF Messages

XJMF Messages are organized into families — Command, Query, etc. (See Section 5.3, “JMF XJMF Message Families” on page 216) — %%and each of these families has Messages for each Message @Type — @Status, KnownDevices, etc. Because it is the convolution of these two that are the unique derived types, the name used in @xsi:type has to be the convolution of the Message Family and Type.

Note XJMF Messages also do not have to be in the default namespace as in the XJDF Node example below.

Example B-3: XJMF: xsi:type

TBD 2.x Example.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="TestSender"
      TimeStamp="2003-11-07T12:15:56Z" MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
    <Query ID="Message_001Q" Type="Events" xsi:type="QueryEvents">
        <NotificationFilter/>
    </Query>
    <Response ID="Message_001R" Type="Events" refID="Q001"
              xsi:type="ResponseEvents">
        <NotificationDef Classes="Error" Type="Barcode"/>
    </Response>
</JMF>

```

Appendix C Supported String and NMOKEN values

C.1 StatusDetails Supported Strings

The *@StatusDetails* Attribute refines the concept of a Job status to be Job specific or a device status to be device specific. The following tables define individual *@StatusDetails* values and map them to the appropriate Job specific state *NodeInfo/@NodeStatus* or device specific state *DeviceInfo/@DeviceStatus*. Note that *NodeInfo/@NodeStatus* = "Setup", "Cleanup" and "Stopped" can include the description of a device with no Job assigned to it.

Note: Although *@StatusDetails* has a datatype of string, it SHOULD be machine readable and thus SHOULD NOT be localized. Localized user data SHOULD be specified in *@DescriptiveName* or Comment elements.

Table C-1: StatusDetails Mapping for Generic Devices (Sheet 1 of 4)

StatusDetails	NodeInfo/ @NodeStatus	DeviceStatus	Description
AbortedBySystem	"Aborted"	"Stopped"	The Job is being or has been aborted by the Device
BreakDown	"Stopped"	"Down"	Breakdown of the device, repair needed.
Calibrating	"Setup"	"Setup"	The Device is calibrating, either manually or automatically.
ControlDeferred	-	"Unknown"	The Machine is not accessible by the Device. Note: JDF/@Status is unknown if the device is not accessible.
CoverOpen	"Stopped"	"Stopped"	One or more covers on the Device are open.
DocumentAccessError	"Aborted"	"Stopped"	The Device could not access one or more documents passed by reference.
DoorOpen	"Stopped"	"Stopped"	One or more doors on the Device are open.
Failure	"Stopped"	"Stopped"	Failure of the device. Requires some maintenance in order to restart the device. "Failure" has specialized subcategories: "PaperJam", "DoubleFeed", "BadFeed", "BadTrim", "ObliqueSheet", "IncorrectComponent", "IncorrectThickness".
Good	"InProgress"	"Running"	Production of products in progress, good copy counter is on, waste copy counter is off
Idling	"Stopped"	"Running"	Device is running, but no products are produced or consumed. Good and waste copy counter are off.
InputTrayMissing	"Stopped"	"Stopped"	One or more input trays are not in the Device

Table C-1: StatusDetails Mapping for Generic Devices (Sheet 2 of 4)

StatusDetails	NodeInfo/ @NodeStatus	DeviceStatus	Description
<i>InterlockOpen</i>	"Stopped"	"Stopped"	One or more interlock devices on the printer are unlocked.
<i>IterationPaused</i>	"Suspended"	-	"Suspended" specifies that at least one iteration cycle has completed but additional iteration cycles MAY still occur.
<i>JobCanceledByOperator</i>	"Aborted"	-	The Job was canceled by the Device operator using "AbortQueueEntry" @Operation of ModifyQueueEntry or means local to the Device.
<i>JobCanceledByUser</i>	"Aborted"	-	The Job was canceled by the owner of the Job using "AbortQueueEntry" @Operation of ModifyQueueEntry.
<i>JobCompletedSuccessfully</i>	"Completed"	-	The Job completed successfully.
<i>JobCompletedWithErrors</i>	"Completed"	-	The Job completed with errors (and possibly warnings too)
<i>JobCompletedWithWarnings</i>	"Completed"	-	The Job completed with warnings.
<i>JobHeld</i>	"Waiting"	-	The Device held the Job that had been waiting (by performing a "HoldQueueEntry" @Operation of ModifyQueueEntry request on a Waiting QueueEntry).
<i>JobHeldOnCreate</i>	"Waiting"	-	The Job was submitted to the queue with the Queue/@Status = "Held", the Job's QueueSubmissionParams/@Held = "true", or JDF/@Activation = "Held".
<i>JobIncoming</i>	"Waiting"	-	The Device is retrieving/accepting document data.
<i>JobResuming</i>	"Waiting"	-	The Device is in the process of moving the Job from a suspended condition to a candidate for processing ("ResumeQueueEntry" @Operation of ModifyQueueEntry).
<i>JobScheduling</i>	"Waiting"	-	The Device is scheduling the Job for processing.

Table C-1: StatusDetails Mapping for Generic Devices (Sheet 3 of 4)

StatusDetails	NodeInfo/ @NodeStatus	DeviceStatus	Description
<i>JobStreaming</i>	"InProgress"	-	Same as " <i>JobIncoming</i> " with the specialization that the Device is processing the document data as it is being received (that is, the Job data is not being spooled, but rather is being processed in chunks by the output device and is being imaged during reception).
<i>JobSuspended</i>	"Suspended"	-	The Device suspended the Job that had been processing (e.g., by performing a " <i>SuspendQueueEntry</i> " @Operation of <i>ModifyQueueEntry</i> request on a Running QueueEntry) and other Jobs can be processed by the Device.
<i>JobSuspending</i>	"InProgress"	"Running"	The Device is in the process of moving the Job from a processing condition to a suspended condition where other Jobs can be processed.
<i>Maintenance</i>	"Stopped"	"Stopped"	Maintenance of the device. " <i>Maintenance</i> " has specialized subcategories: " <i>BlanketChange</i> " and " <i>SleeveChange</i> ".
<i>MissResources</i>	"Stopped"	"Stopped"	Production has been stopped because Resources are missing or unavailable. Waits for new Resources; subcategory of " <i>Pause</i> ".
<i>MovingToPaused</i>	"InProgress"	"Running"	The Device has been paused, but the Machine(s) are taking an appreciable time to stop.
<i>OutputAreaFull</i>	"Stopped"	"Stopped"	One or more output areas are full (e.g., tray, stacker, collator).
<i>OutputTrayMissing</i>	"Stopped"	"Stopped"	One or more output trays are not in the Device
<i>PaperJam</i>	"Stopped"	"Stopped"	Media jam in the device; subcategory of " <i>Failure</i> ".
<i>Pause</i>	"Stopped"	"Stopped"	Machine paused; restart is possible. " <i>Pause</i> " has specialized subcategories: " <i>MissResources</i> " and " <i>WaitForApproval</i> ".

Table C-1: StatusDetails Mapping for Generic Devices (Sheet 4 of 4)

StatusDetails	NodeInfo/ @NodeStatus	DeviceStatus	Description
<i>ProcessingToStopPoint</i>	"InProgress"	"Running"	The requester has issued an " <i>AbortQueueEntry</i> " @Operation of <i>ModifyQueueEntry</i> request or the Device has aborted the Job, but is still performing some actions on the Job until a specified stop point occurs or Job termination/cleanup is completed.
<i>Repair</i>	"Stopped"	"Down"	The device is being repaired after a break down.
<i>ShutDown</i>	"Stopped"	"Down"	Machine stopped (can be switched off), restart requires a run up.
<i>SizeChange</i>	"Setup"	"Setup"	Changing setup for media size.
<i>WaitForApproval</i>	"Stopped"	"Stopped"	Production has been stopped because a necessary approval is still missing, subcategory of " <i>Pause</i> ".
<i>WarmingUp</i>	"Setup"	"Setup"	Device is warming up after power up or power saver mode wake-up.
<i>Waste</i>	"InProgress"	"Running"	Production of products in progress, good copy counter is off, waste copy counter is on.
<i>WasteFull</i>	"Stopped"	"Stopped"	The Device waste receptacle is full.

Table C-2: StatusDetails Mapping for Printing Devices (Sheet 1 of 2)

StatusDetails	NodeInfo/ @NodeStatus	DeviceStatus	Description
<i>BlanketChange</i>	"Stopped"	"Stopped"	Changing of blankets; subcategory of " <i>Maintenance</i> " (e.g., a 'specialization').
<i>BlanketWash</i>	"Cleanup"	"Cleanup"	Washing of the blanket; subcategory of " <i>WashUp</i> ".
<i>CleaningInkFountain</i>	"Cleanup"	"Cleanup"	Cleaning of the ink fountain; subcategory of " <i>WashUp</i> ".
<i>CylinderWash</i>	"Cleanup"	"Cleanup"	Washing of impression cylinders; subcategory of " <i>WashUp</i> ".
<i>DampeningRollerWash</i>	"Cleanup"	"Cleanup"	Washing of the dampening roller; subcategory of " <i>WashUp</i> ".
<i>FormChange</i>	"Setup"	"Setup"	In conventional printing, changing of plates; in direct imaging printing, imaging or re-imaging of plates.
<i>InkRollerWash</i>	"Cleanup"	"Cleanup"	Washing of the inking roller; subcategory of " <i>WashUp</i> ".
<i>PlateWash</i>	"Cleanup"	"Cleanup"	Washing of the plate; subcategory of " <i>WashUp</i> ".

Table C-2: StatusDetails Mapping for Printing Devices (Sheet 2 of 2)

StatusDetails	NodeInfo/ @NodeStatus	DeviceStatus	Description
<i>Processing</i>	<i>"InProgress"</i>	-	Other productive processing (RIP, etc.) is taking place but no final output is being produced. All input data has arrived (not <i>"InProgress"/"JobStreaming"</i> nor <i>"Waiting"/"JobIncoming"</i>).
<i>SleeveChange</i>	<i>"Stopped"</i>	<i>"Stopped"</i>	Changing of sleeves; subcategory for <i>"Maintenance"</i> .
<i>WashUp</i>	<i>"Cleanup"</i>	<i>"Cleanup"</i>	Machine is washed before, during or after production. <i>"WashUp"</i> has specialized subcategories: <i>"BlanketWash"</i> , <i>"CleaningInkFountain"</i> , <i>"CylinderWash"</i> , <i>"DampeningRollerWash"</i> , <i>"InkRollerWash"</i> , or <i>"PlateWash"</i> . <i>"WashUp"</i> is the default which is assumed if <i>@StatusDetails</i> is not specified.
<i>WaitingForMarker</i>	<i>"Suspended"</i> or <i>"Ready"</i>	-	<p>The NodeInfo/@NodeStatus is <i>"Suspended"</i> if the Printing Device models any module prior to the Marker module, otherwise, the NodeInfo/@NodeStatus is <i>"Ready"</i>.</p> <p>The Node is automatically Suspended by the Worker because it is waiting behind other Jobs for the Marker module and the Worker will resume the Node when a Marker module becomes available.</p>
<i>WaitingForReferenceDataCollector</i>	<i>"Suspended"</i> or <i>"Ready"</i>	-	<p>The NodeInfo/@NodeStatus is <i>"Suspended"</i> if the Printing Device models any module prior to the Referenced-Data-Collector module, otherwise, the NodeInfo/@NodeStatus is <i>"Ready"</i>.</p> <p>The Node is automatically Suspended by the Worker because it is waiting behind other Jobs for the Referenced-Data-Collector module and the Worker will resume the Node when a Referenced-Data-Collector module becomes available.</p>
<i>WaitingForRIP</i>	<i>"Suspended"</i> or <i>"Ready"</i>	-	<p>The NodeInfo/@NodeStatus is <i>"Suspended"</i> if the Printing Device models any module prior to the RIP module, otherwise, the NodeInfo/@NodeStatus is <i>"Ready"</i>.</p> <p>The Node is automatically Suspended by the Worker because it is waiting behind other Jobs for a RIP module (process slot) and the Worker will resume the Node when a RIP module becomes available.</p>

Table C-3: StatusDetails Mapping for Postpress Devices

StatusDetails	NodeInfo/ @NodeStatus	DeviceStatus	Description
<i>BadFeed</i>	"Stopped"	"Stopped"	Bad feed on a feeder; subcategory of "Failure".
<i>BadTrim</i>	"Stopped"	"Stopped"	Bad trimmed components; subcategory of "Failure".
<i>DoubleFeed</i>	"Stopped"	"Stopped"	Double feeds on a feeder; subcategory of "Failure".
<i>IncorrectComponent</i>	"Stopped"	"Stopped"	Incorrect components on a feeder; subcategory of "Failure".
<i>IncorrectThickness</i>	"Stopped"	"Stopped"	Incorrect thickness of components; subcategory of "Failure".
<i>ObliqueSheet</i>	"Stopped"	"Stopped"	Oblique Sheets on components; subcategory of "Failure". Oblique Sheets are Sheets or Signatures which are not properly aligned within a pile (e.g., on a gathering or collecting chain).

C.2 ModuleType Supported Strings

The **ModuleStatus** Element (see Table 5-60, “ModuleStatus Element” on page 250) and **VarnishingParams** (see Section 8.109, “VarnishingParams” on page 787) contain a **@ModuleType** Attribute that defines individual modules within a Machine. The following table defines individual **@ModuleType** values.

Table C-4: ModuleType Attribute Values for Conventional Printing Devices

ModuleType	Description
<i>CoatingModule</i>	Unit for coatings, for example, full coating of varnish.
<i>Delivery</i>	Delivery module, unit for gathering the printed Sheets.
<i>Drier</i>	Module for drying the previously printed color or varnish.
<i>ExtensionModule</i>	Unit for extending the distance between modules, for example to increase the distance between the last printing module and the delivery module.
<i>Feeder</i>	Feeder module, feeds the device with paper.
<i>Imaging</i>	Imaging Module in a direct to plate Machine.
<i>Numbering</i>	Numbering unit.
<i>PerfectingModule</i>	Unit for perfecting, reversing device.
<i>PrintModule</i>	Unit for printing a color. Describes one cylinder and one side.

Table C-5: ModuleType Attribute Values for Postpress

ModuleType	Description
<i>BlockPreparer</i>	The Block Preparer prepares the book block for a hardcover book. See Section 6.6.1, “BlockPreparation”.
<i>BoxFolder</i>	The Box Folder folds and glues blanks into folded boxes for packaging. See Section 6.6.2, “BoxFolding”.
<i>CaseMaker</i>	The Case Maker produces the hard case for books. See Section 6.6.5, “CaseMaking”.

Table C-5: ModuleType Attribute Values for Postpress

ModuleType	Description
<i>Caser</i>	The Caser joins the hard cover book case and the book block. (CasingIn). See Section 6.6.6, “CasingIn”.
<i>Chain</i>	Transport chain or conveyer to transport gathered / collected product.
<i>EndSheetGluer</i>	The End-Sheet Gluer merges the front-end sheet, the book block and the back-end sheet together. See Section 6.6.13, “EndSheetGluing”.
<i>Feeder</i>	Feeder module, feeds the device with paper. See Section 6.6.14, “Feeding”.
<i>Gluer</i>	The Gluer applies glue to a component. See Section 6.6.17, “Gluing”.
<i>HeadBandApplicator</i>	The Head Band Applicator applies a head band to the book block. See Section 6.6.18, “HeadBandApplication”.
<i>InkjetPrinter</i>	Prints images or texts on a component. (DigitalPrinting)
<i>Inserter</i>	The Inserter inserts one or more “child” components to one “mother” component. See Section 6.6.20, “Inserting”.
<i>Jacketer</i>	The Jacketer wraps a jacket around a book. See Section 6.6.21, “Jacketing”.
<i>PaperPath</i>	Paper path module, path that paper follows through the Machine.
<i>PressingStation</i>	The Pressing Station presses the cover to the book block.
<i>ShapeCutter</i>	The Shape Cutter produces special shapes like an envelope window or a heart-shaped beer mat. Note that the Shape Cutter Module MAY contain Tools that correspond to the actual dies etc. See Section 6.6.28, “ShapeCutting”.
<i>SpinePreparer</i>	The Spine Preparer prepares the spine of a book for hard and soft cover production. See Section 6.6.31, “SpinePreparation”.
<i>SpineTaper</i>	The Spine Taper applies a tape strip to the spine of a book block. See Section 6.6.32, “SpineTaping”.
<i>Strapper</i>	The Strapper straps a bundle of products. See Section 6.6.36, “Strapping”.
<i>ThreadSealer</i>	The Thread sealer sews and seals a signature at the spine. See Section 6.6.37, “ThreadSealing”.
<i>ThreadSewer</i>	The Thread sewer sews all signatures of a book block together. See Section 6.6.38, “ThreadSewing”.

Table C-6: ModuleType Attribute Values for DigitalPrinting (Sheet 1 of 2)

ModuleType	Description
<i>FarmPrinter</i>	Individual Printer in a printer farm of printers.
<i>Fuser</i>	Fuser module — fuses the toner onto the media.
<i>Marker</i>	Marker module, excluding in-line finishing.
<i>Unpacker</i>	Module that receives and unpacks the ZIP package and fetches the XJDF if it is referenced from the XJMF .

Table C-6: ModuleType Attribute Values for DigitalPrinting (Sheet 2 of 2)

ModuleType	Description
<i>ReferencedDataCollector</i>	Module that fetches data referenced from the XJDF and MAY include data referenced from the PDL. Does not include accepting ZIP, unpacking ZIP, or fetching the XJDF itself.
<i>RIP</i>	RIP module. See Section 6.4.20.1, “RIPing”%##

Table C-7: ModuleType Attribute Values for PrintingUnitWebPath Modules of Web Printing Devices

ModuleType	Description
<i>ChillUnit</i>	Chill unit that chills down the heated printed paper.
<i>ImprintUnit</i>	Printing unit that allows changing plates during production run, doing imprints.
<i>PrintUnit</i>	A Print Unit consists of multiple Print Module units.
<i>RemoisteningModule</i>	Module that can be used for high gloss varnish, remoistened glue, rub-off ink or encapsulated fragrances. The Remoistening Module is located between last printing unit and dryer.
<i>UVCoater</i>	The UV-Coater module applies UV-varnish with subsequent drying in a UV-dryer.

Table C-8: ModuleType Attribute Values for FolderSuperstructureWebPath Modules of Web Printing Devices (Sheet 1 of 2)

ModuleType	Description
<i>CrossCutter</i>	Cuts the Web / ribbon n-times into Sheets and transports the Sheets to inline postpress-equipment
<i>Delivery</i>	Delivers the printed and/or folded Sheets out of the folder
<i>Folder</i>	Module for cutting the collected ribbons into Sheets, in some cases collecting these Sheets, and folding the Sheets (quarter and cross folds)
<i>Former</i>	Module for gathering ribbons and in most instances doing the first fold of the ribbons (quarter fold).
<i>GluingAndSofteningModule</i>	Consists multiple heads, spread out in the press for gluing or/and softening of ribbons or folded Sheets
<i>MoebiusDeinitializer</i>	Used to resolve the infinite loops caused by printing on interleaving surfaces of Möbius banded webs.
<i>PerforatingModule</i>	Module for doing cross, longitudinal or diagonal perforations and die cuts on a Web. Module is placed between Chill Unit and Folder.
<i>PlanoModule</i>	The Plano Module cuts the Web / ribbon into Sheets and stacks the Sheets to a pile
<i>PloughFoldModule</i>	The Plough Fold Module does a quarter fold to ribbons or webs, mostly found in front of a Folder module
<i>Rewinder</i>	Rewinds the printed Web to a Roll.
<i>RibbonCompensator</i>	Controls the Web / ribbons in running direction regarding the cross cut
<i>Slitter</i>	Module for cutting in Machine direction
<i>Stitcher</i>	Stitches folded Sheets together
<i>Superstructure</i>	Module in which a Web will be cut into ribbons and these will be moved to the correct position for folding.

Table C-8: ModuleType Attribute Values for FolderSuperstructureWebPath Modules of Web Printing Devices (Sheet 2 of 2)

ModuleType	Description
<i>TurnerBar</i>	Turns the front side of a Web to the back side and vice versa.
<i>TurnerBarUnit</i>	Turns the front side of a Web to the back side and vice versa in a separate unit.

Table C-9: ModuleType Attribute Values for PostPressComponentPath Modules of Web Printing Devices

ModuleType	Description
<i>BundlingModule</i>	The Bundling Module is used for bundling components
<i>LabelingModule</i>	The Labeling Module is used for labelling a bundle.
<i>PalletizingModule</i>	The Palletizing Module collects the Bundles on a pallet. See Section 6.6.26, “Palletizing” on page 405.
<i>Stacker</i>	Stacks the component to a pile. See Section 6.6.33, “Stacking” on page 408.
<i>Trimmer</i>	Trims the component to its final size. See Section 6.6.39, “Trimming” on page 410.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="WorkflowController"
     TimeStamp="2005-07-25T12:32:48+02:00" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
  <Signal ID="S1" Type="Notification" xsi:type="SignalNotification">
    <Notification Class="Event" JobID="myJobID"
                 TimeStamp="2005-05-25T12:32:48+02:00"
                  Type="Milestone">
      <Comment>All Proofs sent to customer</Comment>
      <Milestone MilestoneType="ProofSent" TypeAmount="24"/>
    </Notification>
  </Signal>
</JMF>

```

C.3 Milestone Values

The following table defines a list of values that are valid for **Content/@ContentStatus** and **Milestone/@MilestoneType**. The column “XJDF Process” specifies the **XJDF@Category** or **XJDF/@Types** that the **Milestone** applies to. “PageStatus” specifies whether the value MAY be used as **Content /@ContentStatus**. “Milestone” specifies whether the value MAY be used as **Milestone/@MilestoneType**. Note that Milestones usually refer to events involving multiple objects, although the **Milestone/@MilestoneType** is specified as a singular. The scope of the **Milestone** is defined by the parent **Notification** Element.

Note: the following symbols are used in the table below:

- **DigDel** means that the files specified in the originating XJDF have arrived in the **XJDF** Process column.
- **Delivery** means **Delivery** in the **XJDF** Process column.
- **ArtDelI** means **DeliveryParams** in the **XJDF** Intent Resource column.
- **DeliveryI** means **DeliveryParams** in the **XJDF** Intent Resource column.
- **ProofingI** means **ContentCheckIntent** in the **XJDF** Intent Resource column

Table C-10: MessageEvents and MilestoneType Values (Sheet 1 of 2)

MessageEvents and MilestoneType Values	XJDF Process	Milestone	Content Status	Description
Accepted	DigDel	—	Yes	The receiver acknowledged that the files are accessible for their destination.
BindingCompleted	—	Yes	Yes	All binding Nodes including packing of the Job have been completed. Postpress Nodes are defined according to Section 6.6, “Postpress Processes”.
BindingInProgress	—	Yes	Yes	At least one of the binding Nodes of the Job is in progress status.
Delivered	DigDel	Yes	Yes	The files were delivered to the destination.
DeviceStopped	All	—	—	The device that executes the Node has been stopped.
DigitalArtArrived	—	Yes	Yes	Digital content has been received.
JobCompletedSuccessfully	All	Yes	Yes	Job completed successfully.
JobCompletedWithErrors	All	Yes	Yes	Job completed with errors.
JobCompletedWithWarnings	All	Yes	Yes	Job completed with warnings.
JobInProgress	All	Yes	Yes	Job is in progress.
PageApproved	—	Yes	Yes	Planned page proofs have been approved.
PageCompleted	—	Yes	Yes	Pages are ready (no further page processing or page proofing required).
PageDeleted	—	—	Yes	Specifies that this, originally planned, page was deleted. For instance, in the past, the page status was “PagePreliminary”. Due to a reduction of the total number of pages, this specific page may have been deleted.
PagePlanned	—	Yes	Yes	Specifies that this page is ready for further processing. Its planning process is finished.
PagePreliminary		Yes	Yes	It is planned to produce this page, but its planning process is not finished yet.
PageProofed	—	Yes	Yes	Planned page proofs have been made
PDLProduced	All	Yes	Yes	Indicates that content data has been produced and is ready for production.
PostPressCompleted	—	Yes	Yes	All Postpress Nodes including packing of the Job have been completed. Postpress Nodes are defined according to Section 6.6, “Postpress Processes”.
PostPressInProgress	—	Yes	Yes	At least one of the Postpress Nodes of the Job is in progress status.

Table C-10: MessageEvents and MilestoneType Values (Sheet 2 of 2)

MessageEvents and MilestoneType Values	XJDF Process	Milestone	Content Status	Description
<i>PrePressCompleted</i>	—	Yes	Yes	All Prepress Nodes of the Job have been completed. Prepress Nodes are defined according to Section 6.4, “Prepress Processes”. In conventional prepress, this is the case when all plates have been made.
<i>PrePressInProgress</i>	—	Yes	Yes	At least one of the Prepress Nodes of the Job is in progress status.
<i>PressCompleted</i>	—	Yes	Yes	All Press Nodes of the Job have been completed. Press Nodes are defined according to Section 6.5, “Press Processes”.
<i>PressInProgress</i>	—	Yes	Yes	At least one of the Press Nodes of the Job is in progress status.
<i>ProofSent</i>	—	Yes	Yes	Planned proofs sent to customer.
<i>ShippingCompleted</i>	Delivery	Yes	Yes	Final product was delivered to the customer or distributors.
<i>ShippingInProgress</i>	Delivery	Yes	Yes	Final product is being shipped.
<i>SurfaceApproved</i>	—	Yes	Yes	Planned imposition proofs have been approved.
<i>SurfaceAssigned</i>	—	Yes	Yes	Surfaces have their corresponding pages assigned (e.g., could be proofed).
<i>SurfaceCompleted</i>	—	Yes	Yes	Planned surfaces are ready (i.e., plates could be made).
<i>SurfaceProofed</i>	—	Yes	Yes	Planned imposition proofs have been made.

C.4 Input Tray and Output Bin Names

Location/@LocationName MAY also be used to specify a *@Location* within a device (e.g., a paper tray). When specifying paper trays, the following locations are predefined. When specifying input paper trays (indicated with “I”) and/or output bins (indicated with “O”), the following values for *Location/@LocationName* locations are predefined. When specifying input tray names, the following values for *Location/@LocationName* are suggested. The input tray names that specify a position (e.g., Top) are identified by an asterisk (*). These positional input tray names SHOULD NOT be used if devices are clustered because the position of the input tray might not be the same for all of the devices in the cluster. (See Section 8.1.1, “Location” on page 519 for more details on the use of Location.)

Table C-11: Input Tray and Output Bin Names (Sheet 1 of 3)

Name	I/O	Description
<i>AnyLargeFormat</i>	IO	The location that holds larger format media with one dimension larger than 11 inches. The media dimensions SHALL be specified. “ <i>AnyLargeFormat</i> ” is defined for a PPD.
<i>AnySmallFormat</i>	IO	The location that holds smaller format media. The media dimensions SHALL be specified. “ <i>AnySmallFormat</i> ” is defined for a PPD.
<i>AutoSelect</i>	IO	The location that the device selects based on the Media specification.
<i>Back</i>	IO*	The value “ <i>Rear</i> ” is analogous; “ <i>Rear</i> ” SHOULD be used instead when possible.

Table C-11: Input Tray and Output Bin Names (Sheet 2 of 3)

Name	I/O	Description
<i>Booklet</i>	O	The bin where the Device places booklets.
<i>Bottom</i>	IO*	The bin that, when facing the device, can best be identified as ‘bottom’.
<i>BypassTray</i>	I	The input tray used to handle odd or special papers. MAY be used to specify the input tray that is used for inserts Sheets that are not to be imaged.
<i>BypassTray-N</i>	I	The input tray used to handle odd or special papers. MAY be used to specify the input tray that is used for inserts Sheets that are not to be imaged. N = ‘1’, ‘2’, ...
<i>Cassette</i>	IO	The value “ <i>Tray-N</i> ” is analogous; “ <i>Tray-N</i> ” SHOULD be used instead when possible.
<i>Continuous</i>	IO	The location to handle continuous media (i.e., continuously connected Sheets).
<i>Disc</i>	IO	The location to handle CD or DVD discs to be printed on.
<i>Disc-N</i>	IO	The location to handle CD or DVD discs to be printed on. N = ‘1’, ‘2’, ...
<i>Envelope</i>	IO	The location that is to contain envelopes.
<i>Envelope-N</i>	IO	The location that is to contain envelopes. N = ‘1’, ‘2’, ...
<i>FaceDown</i>	O	The bin that can best be identified as ‘face down’ with respect to the device.
<i>FaceUp</i>	O	The bin that can best be identified as ‘face up’ with respect to the device.
<i>FitMedia</i>	O	Requests the device to select a bin based on the size of the media.
<i>Front</i>	IO*	The location that, when facing the device, can best be identified as ‘front.’
<i>InsertTray</i>	I	The input tray that can best be identified as ‘insert tray.’ Used to specify the input tray that is used for inserts Sheets (insert Sheets are never imaged).
<i>InsertTray-N</i>	I	The input tray that can best be identified as ‘insert tray-1’, ‘insert tray-2’, ... etc. Used to specify the input tray that is used for inserts Sheets (insert Sheets are never imaged).
<i>LargeCapacity</i>	IO	The bin that can best be identified as the ‘large capacity’ bin (in terms of the number of Sheets) with respect to the device.
<i>LargeCapacity-N</i>	IO	The location that can best be identified as the ‘large capacity-1’, ‘large-capacity-2’, ... etc., input tray (in terms of the number of Sheets) with respect to the device.
<i>Left</i>	IO*	The bin that, when facing the device, can best be identified as ‘left.’
<i>Lower</i>	IO*	The value “ <i>Bottom</i> ” is analogous; “ <i>Bottom</i> ” SHOULD be used instead when possible.
<i>Mailbox-N</i>	O	The Job will be output to the bin that is best identified as “Mailbox #1”, “Mailbox #2”, etc.
<i>Main</i>	IO	The value “ <i>LargeCapacity</i> ” is analogous; “ <i>LargeCapacity</i> ” SHOULD be used instead when possible.
<i>Middle</i>	IO*	The bin that, when facing the device, can best be identified as “middle”.
<i>MyMailbox</i>	O	The Job will be output to the bin that is best identified as “My Mailbox”
<i>PostMarkerInserter</i>	I	The input tray that is downstream of the marking engine and allows the user to pass media through a non-marking paper path for covers and/or inserts.
<i>Rear</i>	IO*	The bin that, when facing the device, can best be identified as “rear”.
<i>Right</i>	IO*	The bin that, when facing the device, can best be identified as “right”.
<i>Roll</i>	IO	The location to handle Web-Fed media.

Table C-11: Input Tray and Output Bin Names (Sheet 3 of 3)

Name	I/O	Description
<i>Roll-N</i>	IO	The Nth location to handle the Nth Web-Fed media.
<i>Side</i>	IO*	The bin that, when facing the device, can best be identified as “side”.
<i>Stacker-N</i>	O	The Job will be output to the bin that is best identified as “Stacker #1”, “Stacker #2”, etc.
<i>Top</i>	IO*	The bin that, when facing the device, can best be identified as “top”.
<i>Tray</i>	IO	The location for a single tray device.
<i>Tray-N</i>	IO	The Job will be output to the tray that is best identified as “Tray #1”, “Tray #2”, etc.
<i>Upper</i>	IO*	The value " <i>Top</i> " is analogous; " <i>Top</i> " SHOULD be used instead when possible.

Appendix D Supported Error Codes in XJMF and Notification Elements

The following list defines the standard *ReturnCode* for messaging. The ID numbers are decimal. Error Messages below 100 are reserved for protocol errors. Error messages above 100 are used for Device and Controller errors and error messages above 200 for Job and pipe specific errors.

Table D-1: Return codes for XJMF (Sheet 1 of 2)

ReturnCode	Description
0	Success
1 – 99	Protocol errors
1	General error
2	Internal error
3	XML parser error (e.g., if a ZIP file is sent to an a Device that does not support ZIP packaging).
4	XML validation error
5	Query Message/Command Message not implemented
6	Invalid parameters
7	Insufficient parameters
8	Device not available (Controller exists but not the Device or queue)
9	Message incomplete.
10	Message Service is busy.
13	Reliable Signals not supported. Subscription denied.
100 – 199	Device and Controller errors
100	Device not running
101	Device incapable of fulfilling request (e.g., a RIP that has been asked to cut a Sheet).
102	No executable Node exists in the XJDF
103	@ <i>JobID</i> not known by Controller
104	@ <i>JobPartID</i> not known by Controller
105	Queue entry not in queue
106	Queue request failed because the queue entry is already executing
107	The queue entry is already executing. Late change is not accepted
108	Selection or applied filter results in an empty list
109	Selection or applied filter results in an incomplete list. A buffer cannot provide the complete list queried for.
110	Queue request of a Job submission failed because the requested completion time of the Job cannot be fulfilled.
111	Subscription request denied.
112	Queue request failed because the Queue is " <i>Closed</i> " or " <i>Blocked</i> " and does not accept new entries.
113	Queue entry is already in the resulting status.
114	QueueEntry/@ <i>Status</i> is already " <i>PendingReturn</i> ", " <i>Completed</i> " or " <i>Aborted</i> " and therefore does not accept changes.

Table D-1: Return codes for XJMF (Sheet 2 of 2)

ReturnCode	Description
115	Queue entry is not running.
116	Queue entry already exists. Used when a <code>QueueEntry</code> with identical <code>@JobID</code> , <code>@JobPartID</code> and <code>Part</code> already exists.
120	Cannot access referenced URL. URI Reference cannot be resolved. Used when a referenced entity (e.g., a <code>JDF</code> in a <code>SubmitQueueEntry</code> cannot be found).
121	Unknown <code>DeviceID</code> . No Device is known with the <code>DeviceID</code> specified.
130	Ganging is not supported. A gang Job has been submitted to a queue that does not support ganging.
131	<code>GangName</code> not known. A Job has been submitted with an unknown <code>GangName</code> .
140	Resource Command rejected
141	Resource Command partially rejected
200 – ...	Job and pipe specific errors
200	Invalid Resources
201	Insufficient Resource parameters
202	<code>PipeID</code> unknown
203	Unlinked Resource.
204	Could not create new <code>JDF</code> Node.

Appendix E Color Adjustment Attribute Description and Usage

This appendix describes several alternative usages of some Attributes in the `ColorCorrectionOp` Element (see `ColorCorrectionParams/ColorCorrectionOp` in Section 8.24, “`ColorCorrectionParams`” on page 583). that are intended to allow simple, late-in-the-workflow, minor adjustments to the overall color appearance of a Job or portions of a Job.

Note: These color adjustments are not available in any Intent Resource, such as `ColorIntent`. In order to request such adjustment in a Product Intent Job ticket supplied to a print provider, attach to a Product Intent Node an incomplete `ColorCorrection` Process with a `ColorCorrectionParams` Resource specifying the requested `ColorCorrectionOp` Element Attributes.

E.1 Adjustment Using Direct Attributes

This section describes the following Attributes that provide direct adjustments to various aspect of the color space:

Table E-1: Attributes for Color Space Adjustment

Attribute Name	Allowed Value Range
<code>AdjustCyanRed</code>	-100 to +100
<code>AdjustMagentaGreen</code>	-100 to +100
<code>AdjustYellowBlue</code>	-100 to +100
<code>AdjustContrast</code>	-100 to +100
<code>AdjustHue</code>	-180 to +180
<code>AdjustLightness</code>	-100 to +100
<code>AdjustSaturation</code>	-100 to +100

These Attributes can be applied at a point where an abstract profile would be applied (following any abstract profiles used) in the order: `@AdjustLightness`, `@AdjustContrast`, `@AdjustSaturation`, `@AdjustHue`, `{@AdjustCyanRed/@AdjustMagentaGreen/@AdjustYellowBlue}`. The operation of each adjustment Attribute is described in relation to colors expressed in the L*a*b* connection color space (with L* expressed on a scale of 0 to 100).

Note: in the C-language-like assignment statements below, the variables L, a and b are used to represent values of the L*, a* and b* channels to avoid ambiguity with the “*” used to denote multiplication in these statements.

- `@AdjustLightness` offsets the L* channel. [L += `@AdjustLightness`]
- `@AdjustContrast` scales the L* channel about mid-scale (where L = 50).
[L = 50 + (L - 50) * (`@AdjustContrast` / 100 + 1)]
- `@AdjustSaturation` scales the a* and b* channels about zero. [a *= (`@AdjustSaturation` / 100 + 1)] and [b *= (`@AdjustSaturation` / 100 + 1)]

`@AdjustCyanRed`, `@AdjustMagentaGreen` and `@AdjustYellowBlue` offset the colors in the a*b* plane along the respective color vector. Lightness (L*) is not changed. Positive values offset towards red, green or blue, and negative values offset towards cyan, magenta or yellow. The adjustment vectors are aligned with the standard SWOP inks. When adjusting Device colors, these adjustments can be approximated by offsets along the vectors of the actual ink colors being used. The angles and unit vectors for SWOP inks (from the CGATS TR001 print characterization) are:

	Red-cyan	Green-Magenta	Blue-yellow
Angle	-129.9	-5.3	94.5

a*	0.641	-0.996	0.078
-----------	-------	--------	-------

b*	0.767	0.092	-0.997
-----------	-------	-------	--------

So

$$\begin{aligned} \mathbf{a}^* &+= 0.641 * @AdjustCyanRed \\ &\quad - 0.996 * @AdjustMagentaGreen \\ &\quad + 0.078 * @AdjustYellowBlue \end{aligned}$$

$$\begin{aligned} \mathbf{b}^* &+= 0.767 * @AdjustCyanRed \\ &\quad + 0.092 * @AdjustMagentaGreen \\ &\quad - 0.997 * @AdjustYellowBlue \end{aligned}$$

@AdjustHue offsets the hue angle value when the colors have been transformed to the CIE- L* C* H* (luminance, chroma and hue) color space from the L*a*b* connection color space. The **@AdjustHue** angle is expressed in degrees.

Note: in the C-language-like assignment statements below, the variables L, a and b are used to represent values of the L*, a* and b* channels to avoid ambiguity with the “*” used to denote multiplication in these statements.

- $a = a * \cos(@AdjustHue) - b * \sin(@AdjustHue)$
- $b = a * \sin(@AdjustHue) + b * \cos(@AdjustHue)$

E.2 Adjustment using ICC Profile Attributes

This section describes two alternatives to the direct color adjustment Attributes providing adjustments of the same nature using ICC profiles. The ICC profile approach provides a standard mechanism for applying a set of multi-dimensional adjustments with a single operation. The ICC profile approach also has an advantage in that it minimizes algorithm and interpretation dependency on the receiving end.

E.3 Adjustment using an ICC Abstract Profile Attribute

A color adjust can be encapsulated in an ICC abstract profile that is applied in ICC Profile Connection Space (PCS). The **FileSpec** Resource of the **ColorCorrectionOp** Element with the **@ResourceUsage** Attribute set to "*AbstractProfile*" references an ICC profile to be used in this manner.

E.4 Adjustment using an ICC DeviceLink Profile Attribute

A color adjust can be encapsulated in an ICC DeviceLink profile that is applied in Device space. The **FileSpec** Resource of the **ColorCorrectionOp** Element with the **@ResourceUsage** Attribute set to "*DeviceLinkProfile*" references an ICC profile to be used in this manner.

Appendix F North American and Japanese Media Weight Explained

In North America and Japan, each grade of paper has one basic size used to compute its basis weight per ream. For example, Bond basic size is 17" x 22" and Shiroku-ban basic size is 788 mm x 1091 mm.

F.1 North American Media Weight

In North America, a paper's basis weight is the weight of five hundred Sheets of its basic size. For example, if five hundred 25" x 38" Sheets of offset paper weigh 60 pounds, it is called 60# offset. Paper mills outside of North America use the metric system to designate paper weight. The basis weight of foreign papers is grams per square meter (g/m²) known as the Sheet's grammage. Papers made to metric standards don't convert to basis weights familiar to North Americans. For example, 100 g/m² equals a basis weight of 67.5. Following is the English/grammage conversion formula:

$$\text{Basis Weight (lb.)} \times (1406.5 / \text{Square inches in basic size}) = \text{grams per square meter}$$

For example, the grammage of 65 lb. cover stock when the cover is 20 x 26 can be calculated as follows:

$$65 \times (1406.5 / (20 \times 26)) = 65 \times 2.70 = 176 \text{ g/m}^2$$

The following table defines the basic sizes and the factor that @USWeight is multiplied by to calculate @Weight for various stock types. Stock type is specified in **Media/@StockType** or **MediaIntent/@StockType**.

Table F-1: Conversion Factor from USWeight (lbs) to Weight (g/m²)

Stock Type	Basis size in Inches	Weight / USWeight	Equivalent
"Bond"	17" x 22"	3.76	"Ledger", "Manifold"
"Book"	25" x 38"	1.48	"Bible", "Coated", "Offset", "Text"
"Bristol"	22½" x 28½"	2.19	
"Cover"	20" x 26"	2.70	
"Index"	25½" x 30½"	1.81	
"Newsprint"	24" x 36"	1.63	"Tag"

In the following table, the right columns of each column pair list common basis weights for North American papers while the left columns list their corresponding grammage. The rows are ordered by grammage. Basis weights for bond, book, cover and other grades of papers are computed using different basic sizes, so the progression of weights down the right columns is untidy.

Table F-2: Grammage Equivalents for Common (US) Basis Weights (Sheet 1 of 2)

Grammage (g/m ²)	Basis Weight	Grammage (g/m ²)	Basis Weight
30	20# Book	150	40# Ledger
34	9# Manifold	152	60# Cover
36	24# Book	163	90 # Index
44	30# Book	163	100 # Tag
45	12# Manifold	175	80# Bristol
49	13# Bond	176	65# Cover
49	33# Book	178	120# Book
52	35# Book	197	90# Bristol
59	40# Book	199	110# Index

Table F-2: Grammage Equivalents for Common (US) Basis Weights (Sheet 2 of 2)

Grammage (g/m²)	Basis Weight	Grammage (g/m²)	Basis Weight
60	16# Bond	204	125# Tag
67	45# Bond	216	80# Cover
74	50# Book	219	100# Bristol
75	20# Bond	244	150# Tag
81	55# Book	253	140# Index
89	60# Book	263	120# Bristol
90	24# Bond	270	100# Cover
104	70# Book	285	175# Tag
105	28# Ledger	307	140# Bristol
108	40# Cover	307	170# Index
118	80# Book	325	200# Tag
120	32# Ledger	350	160# Bristol
133	90# Book	352	130# Cover
135	36# Ledger	394	180# Bristol
135	50# Cover	398	220# Index
147	67# Bristol	407	250# Tag
148	100# Book	438	200# Bristol
		488	300# Tag

F.2 Japanese Media Weight

In Japan, a paper's basis weight is the weight of 1000 Sheets of its basic size and ream weights are given in kg.

The following table is originally published by EDS Inc., Editorial & Design Services at <http://www.edsebooks.com/paper/jpaper.html>. For more help with grammage and basis weight conversion, see also Basis Weight and Grammage Conversion Tables at <http://home.inter.net/eds/paper/grammage.html>.

Following is the Japanese/grammage conversion formula:

$$\text{Basis Weight (kg)} / \text{Basic Size (m}^2\text{)} = \text{grams per square meter}$$

For example, the grammage of 70 kg Shiroku-ban stock when the size is 0.788 x 1.091 can be calculated as follows:

$$70 / (0.788 \times 1.091) = 81.4 \text{ g/m}^2$$

In the table below, trade-sheet size is given in mm.

Table F-3: Japanese Media Weight

Paper Grade *	Shiroku-ban 788 x 1091	JIS B-ban 765 x 1085	Kiku-ban 636 x 939	JIS A-ban 625 x 880	Grammage (g/m ²)
上質紙 Joushitsushi	40	--	--	--	46.5
	45	--	31	20.5	52.3
	55	53	38	35	64.0
	70	67.5	48.5	44.5	81.4
	90	--	62.5	47.5	104.7
	110	--	71.5	70.5	127.9
	135	--	93.5	80.5	157.0
	180	--	--	--	209.3
<hr/>					
中質紙 Chuushitsushi	--	45	--	30	54.2
	--	55	--	36.5	66.3
<hr/>					
アート紙 Aatoshi	73	70.5	50.5	46.5	84.9
	90	87	62.5	57.5	104.7
	110	106	76.5	70.5	127.9
	135	130.5	93.5	86.5	157.0
<hr/>					
マシンコート紙 Mashinkootoshi	63	61	--	--	73.3
	68	65.6	47	43.5	79.1
	73	70.5	50.5	46.5	84.9
	90	87	62.5	57.5	104.7
	110	106	76.5	70.5	127.9
	135	130.5	93.5	86.5	157.0
<hr/>					
アートポスト紙 Aatoposutoshi	180	--	125	--	209.3
	200	--	139	--	232.6
	220	--	153	--	255.0

* The following describes the five paper grades in the above table:

- **上質紙 Joushitsushi** (“top-quality paper”) contains 100% chemical pulp;
- **中質紙 Chuushitsushi** (“medium-quality paper”) contains a minimum of 70% chemical pulp;
- **アート紙 Aatoshi** (“art paper”) is machine coated paper, available in top quality and medium quality (Joushitsu and Chuushitsu);
- **マシンコート紙 Mashinkootoshi** (“machine coated paper”), also called Kootoshi (コート紙), is machine coated paper given only a thin coat of clay;
- **アートポスト紙 Aatoposutoshi** (“art-post paper”) is cover stock coated on one side.

Appendix G Media Sizes

The following table defines a set of named media sizes as defined by [PPD].

Implementation Remark

Since Media sizes may be real numbers, comparison of Media sizes SHOULD take into account certain rounding errors. For example, different Media sizes SHOULD be considered equal when all numbers are the same within a range of 5 point.

Key for Notes

- I — Size is defined by ISO standards, including [ISO216:1975].
- J — Size is defined by JIS standards [JIS P0138:1998]
- E — This is an envelope size [ISO269:1985].

Table G-1: Media Sizes (Sheet 1 of 4)

Media Size	Size in Points	Size in Millimeters	Size in Inches	Notes
A0	2384 x 3370	841 x 1189	33.11 x 46.81	I, J
A1	1684 x 2384	594 x 841	23.39 x 33.11	I, J
A2	1191 x 1684	420 x 594	16.54 x 23.39	I, J
A3	842 x 1191	297 x 420	11.69 x 16.54	I, J
A3Extra	913 x 1262	322 x 445	12.67 x 17.52	
A4	595 x 842	210 x 297	8.27 x 11.69	I, J
A4Extra	667 x 914	235.5 x 322.3	9.27 x 12.69	
A4Plus	595 x 936	210 x 330	8.27 x 13	
A4Tab	638 x 842	225 x 297	8.86 x 11.69	
A5	420 x 595	148 x 210	5.83 x 8.27	I, J
A5Extra	492 x 668	174 x 235	6.85 x 9.25	
A6	297 x 420	105 x 148	4.13 x 5.83	I, J
A7	210 x 297	74 x 105	2.91 x 4.13	I, J
A8	148 x 210	52 x 74	2.05 x 2.91	I, J
A9	105 x 148	37 x 52	1.46 x 2.05	I, J
A10	73 x 105	26 x 37	1.02 x 1.46	I, J
AnsiC	1224 x 1584	431.8 x 558.8	17 x 22	
AnsiD	1584 x 2448	558.8 x 863.6	22 x 34	
AnsiE	2448 x 3168	863.6 x 1118	34 x 44	
ARCHA	648 x 864	228.6 x 304.8	9 x 12	
ARCB	864 x 1296	304.8 x 457.2	12 x 18	
ARCHC	1296 x 1728	457.2 x 609.6	18 x 24	
ARCHD	1728 x 2592	609.6 x 914.4	24 x 36	
ARCHE	2592 x 3456	914.4 x 1219	36 x 48	
B0	2920 x 4127	1030 x 1456	40.55 x 57.32	J
B1	2064 x 2920	728 x 1030	28.66 x 40.55	J
B2	1460 x 2064	515 x 728	20.28 x 28.66	J

Table G-1: Media Sizes (Sheet 2 of 4)

Media Size	Size in Points	Size in Millimeters	Size in Inches	Notes
B3	1032 x 1460	364 x 515	14.33 x 20.28	J
B4	729 x 1032	257 x 364	10.12 x 14.33	J
B5	516 x 729	182 x 257	7.17 x 10.12	J
B6	363 x 516	128 x 182	5.04 x 7.17	J
B7	258 x 363	91 x 128	3.58 x 5.04	J
B8	181 x 258	64 x 91	2.52 x 3.58	J
B9	127 x 181	45 x 64	1.77 x 2.52	J
B10	91 x 127	32 x 45	1.26 x 1.77	J
C4	649 x 918	229 x 324	9.02 x 12.75	I, E
C5	459 x 649	162 x 229	6.38 x 9.02	I, E
C6	323 x 459	114 x 162	4.51 x 6.38	I, E
Comm10	297 x 684	104.8 x 241.3	4.125 x 9.5	E
DL	312 x 624	110 x 220	4.33 x 8.66	I, E
DoublePostcard	567 x 419	200 x 148	7.87 x 5.83	
Env9	279 x 639	98.4 x 225.4	3.875 x 8.875	E
Env10	297 x 684	104.8 x 241.3	4.125 x 9.5	E
Env11	324 x 747	113.3 x 263.5	4.5 x 10.375	E
Env12	342 x 792	120.7 x 279.4	4.75 x 11	E
Env14	360 x 828	127 x 292.1	5 x 11.5	E
EnvC0	2599 x 3676	917 x 1297	36.10 x 51.06	I, E
EnvC1	1837 x 2599	648 x 917	25.51 x 36.10	I, E
EnvC2	1298 x 1837	458 x 648	18.03 x 25.51	I, E
EnvC3	918 x 1296	324 x 458	12.75 x 18.03	I, E
EnvC4	649 x 918	229 x 324	9.02 x 12.75	I, E
EnvC5	459 x 649	162 x 229	6.38 x 9.02	I, E
EnvC6	323 x 459	114 x 162	4.51 x 6.38	I, E
EnvC65	324 x 648	114 x 229	4.51 x 9	E
EnvC7	230 x 323	81 x 113	3.19 x 4.49	I, E
EnvChou3	340 x 666	120 x 235	4.72 x 9.25	E
EnvChou4	255 x 581	90 x 205	3.54 x 8	E
EnvDL	312 x 624	110 x 220	4.33 x 8.66	I, E
EnvInvite	624 x 624	220 x 220	8.66 x 8.66	E
EnvISOB4	708 x 1001	250 x 353	9.84 x 13.9	E
EnvISOB5	499 x 709	176 x 250	6.9 x 9.8	E
EnvISOB6	499 x 354	176 x 125	6.9 x 4.9	E
EnvItalian	312 x 652	110 x 230	4.33 x 9	E
EnvKaku2	680 x 941	240 x 332	9.45 x 13	E
EnvKaku3	612 x 785	216 x 277	8.5 x 10.9	E

Table G-1: Media Sizes (Sheet 3 of 4)

Media Size	Size in Points	Size in Millimeters	Size in Inches	Notes
EnvMonarch	279 x 540	98.43 x 190.5	3.875 x 7.5	E
EnvPersonal	261 x 468	92.08 x 165.1	3.625 x 6.5	E
EnvPRC1	289 x 468	102 x 165	4 x 6.5	E
EnvPRC2	289 x 499	102 x 176	4 x 6.9	E
EnvPRC3	354 x 499	125 x 176	4.9 x 6.9	E
EnvPRC4	312 x 590	110 x 208	4.33 x 8.2	E
EnvPRC5	312 x 624	110 x 220	4.33 x 8.66	E
EnvPRC6	340 x 652	120 x 230	4.7 x 9	E
EnvPRC7	454 x 652	160 x 230	6.3 x 9	E
EnvPRC8	340 x 876	120 x 309	4.7 x 12.2	E
EnvPRC9	649 x 918	229 x 324	9 x 12.75	E
EnvPRC10	918 x 1298	324 x 458	12.75 x 18	E
EnvYou4	298 x 666	105 x 235	4.13 x 9.25	E
Executive	522 x 756	184.2 x 266.7	7.25 x 10.5	
FanFoldGerman	612 x 864	215.9 x 304.8	8.5 x 12	
FanFoldGermanLegal	612 x 936	215.9 x 330	8.5 x 13	
FanFoldUS	1071 x 792	377.8 x 279.4	14.875 x 11	
Folio	595 x 935	210 x 330	8.27 x 13	
ISOB0	2835 x 4008	1000 x 1414	39.37 x 55.67	I
ISOB01	2004 x 2835	707 x 1000	27.83 x 39.37	I
ISOB2	1417 x 2004	500 x 707	19.68 x 27.83	I
ISOB3	1001 x 1417	353 x 500	13.90 x 19.68	I
ISOB4	709 x 1001	250 x 353	9.84 x 13.90	I
ISOB5	499 x 709	176 x 250	6.9 x 9.8	I
ISOB5Extra	569 x 782	201 x 276	7.9 x 10.8	
ISOB6	354 x 499	125 x 176	4.92 x 6.93	I
ISOB7	249 x 354	88 x 125	3.46 x 4.92	I
ISOB8	176 x 249	62 x 88	2.44 x 3.46	I
ISOB9	125 x 176	44 x 62	1.73 x 2.44	I
ISOB10	88 x 125	31 x 44	1.22 x 1.73	I
Ledger	1224 x 792	431.8 x 279.4	17 x 11	
Legal	612 x 1008	215.9 x 355.6	8.5 x 14	
LegalExtra	684 x 1080	241.3 x 381	9.5 x 15	
Letter	612 x 792	215.9 x 279.4	8.5 x 11	
LetterExtra	684 x 864	241.3 x 304.8	9.5 x 12	
LetterPlus	612 x 913	215.9 x 322.3	8.5 x 12.69	
LetterTabThreeEighthsInch	639 x 792	225.4 x 279.4	8.875 x 11	
LetterTabHalfInch	648 x 792	228.6 x 279.4	9 x 11	

Table G-1: Media Sizes (Sheet 4 of 4)

Media Size	Size in Points	Size in Millimeters	Size in Inches	Notes
LetterTabFiveEighthsInch	657 x 792	231.8 x 279.4	9.125 x 11	
Monarch	279 x 540	98.43 x 190.5	3.875 x 7.5	E
Postcard	284 x 419	100 x 148	3.94 x 5.83	
PRC16K	414 x 610	146 x 215	5.75 x 8.5	
SRA3	907 x 1276	320 x 450	12.60 x 17.72	

Appendix H **MimeType and MimeTypeVersion Attributes**

This appendix lists examples values for the following Attributes of the **FileSpec** Resource: **@MimeType** and **@MimeTypeVersion**. The preferred file name extension is also indicated for use in the **FileSpec/@URL** Attribute., respectively.

The listing is intended to be exhaustive for the most likely document formats that are routinely used in **XJDF** applications. However, other document formats and other combinations of the listed document formats can be used as well. When these format standards are revised with new version numbers, they MAY be used and SHOULD follow the patterns established in the following tables.

Many **@MimeTypeVersion** values are taken from the *Printer MIB* [RFC1759] by using the a language (e.g., PS, PCL, etc.) as a prefix followed by the level or version defined for **prtInterpreterLangLevel** separated by a “/” character (e.g., “PS/3” for PostScript Level 3). For file formats not in the *Printer MIB*, the prefix is the common acronym for the format with “/” changed to “-” so that the prefix always ends with the first “/” (e.g. “DCS/2.0” for DCS version 2.0 and “TIFF-IT/BL/P1:1998” for TIFF/IT — Binary Line art image data — profile 1).

Table H-1 lists the **@MimeType** values that are MIME Media Types registered with IANA (as opposed to file types which are not registered with IANA) in alphabetical order, as well as possible **@MimeTypeVersion** values. A blank **@MimeTypeVersion** table entry indicates that there is no recognized version number for the **@MimeType**. Table H-1 also lists the associated RECOMMENDED file name extensions commonly used by **XJDF** applications. Note: According to [RFC2046] the initial set of MIME media types start with the substrings: “application/”, “audio/”, “image/”, “message/”, “model/”, “multipart/”, “text/” or “video/”. File Types will not start with these strings. The **@Compression** values that do have a corresponding IANA MIME type are also listed, so that a file that is so compressed or encoded has an appropriate **@MimeType** value for the file, as shown below.

Table H-1: MimeType Attribute Values (IANA Registered) (Sheet 1 of 3)

MimeType	Sample MimeType- Version	File Extension	Description [iana-mt] indicates IANA registration
application/mac-binhex40	HQX/4.0	.hqx	Macintosh BinHex 4.0 7-bit encoding [RFC1741] Note: BinHex encoding converts an 8-bit file into a 7-bit format [RFC1741], similar to Uuencoding. BinHex format preserves file Attributes, as well as Macintosh resource forks, and includes CRC (Cyclic Redundancy Check) error-checking. This encoding method works on any type of file, including formatted word processing and spreadsheet files, graphics files and even executable files (i.e., programs or applications). Note: BinHex is not to be confused with MacBinary encoding, which is an 8-bit format.
application/msword	MSWORD/XP	.doc	Microsoft Word
application/pdf	PDF/1.6, PDF/X-3:2003	.pdf	Adobe Portable Document Format [PDF1.6] and Portable Document Format (PDF) PDF/X-3 [ISO15930-6:2003]
application/postscript	PS/3	.ps	Adobe PostScript™ See [RFC2045] and [RFC2046]

Table H-1: MimeType Attribute Values (IANA Registered) (Sheet 2 of 3)

MimeType	Sample MimeType- Version	File Extension	Description [iana-mt] indicates IANA registration
application/vnd.cip4-jdf+xml	JDF 1.5	.jdf	CIP4 Job Definition Format (JDF) version 1.5.
application/vnd.cip4-jmf+xml	JMF 1.5	.jmf	CIP4 Job Definition Format (JDF) version 1.5 (See Job Messaging Format).
application/vnd.cip4-ptk+xml	PrintTalk 1.5	.ptk	CIP4 PrintTalk version 1.5
application/vnd.cip3-ppf	PPF/3.0	.ppf	CIP3 Print Production Format (PPF) version 3.0, 1998 [PPF]
application/vnd.hp-PCL	PCL/X	.pcl	Hewlett Packard Printer Control Language (PCL™)
application/vnd.iccprofile		.icc .icm	International Color Consortium (ICC) File Format for Color Profiles taken from the binary coded decimal Profile Header Profile Version Number field (bytes 8 through 11) [ICC.1]
application/vnd.podi-ppml+xml	PPML/2.1	.ppml	Personalized Print Markup Language [PPML]
application/vnd.Quark.QuarkXPress	XPress/6.0	.qxd .qxt .qwd .qwt .qxl .qxb	QuarkXPress [Quark]
application/zip		.zip	ZIP packaging — The actual compression used for each file in a ZIP package is stored in the ZIP package as metadata for each file. Therefore, the FileSpec/@Compression Attribute for the contained file MAY use any @Compression value, including "None", "Compress", "Gzip" and "ZLIB".
image/jpeg		.jpeg .jpg	JPEG See [RFC2045] and [RFC2046]. Note: image/jpeg is really an image format, not a file format. JFIF and EXIF are file formats that contain image/jpeg image format data, and some applications have their own formats that are similar to JFIF and EXIF but which are proprietary. None the less, the "image/jpeg" @MimeType value is used to identify these file types.

Table H-1:MimeType Attribute Values (IANA Registered) (Sheet 3 of 3)

MimeType	Sample MimeType- Version	File Extension	Description [iana-mt] indicates IANA registration
image/tiff	tiff/6.0	.tiff .tif	Tag Image File Format [RFC3302] Note: The image/tiff MIME @Media Type is assumed to be TIFF Revision 6.0 as defined in detail by Adobe in [TIFF6]. TIFF/IT is a different MIME type.
x-world/x-vrml			

Table H-2 lists the *@MimeType* values that are file types assigned by CIP4 (as opposed to MIME Media Types which are registered with IANA) and possible *@MimeTypeVersion* values commonly used in XJDF applications. A blank *@MimeTypeVersion* table entry indicates that there is no recognized version number for the *@MimeType*. Table H-2 also lists associated RECOMMENDED file name extensions values. A blank file extension column entry indicates that there is no recognized file name extension for the *@MimeType*. The *@Compression* values that do not have a corresponding IANA MIME type are also assigned a file type value, so that a file that is so compressed or encoded has an appropriate *@MimeType* value for the file, as shown in the table below.

Table H-2: mimeType and File Type Combinations (Sheet 1 of 3)

MimeType	File Extension	Description [iana-mt] indicates IANA registration
Base64	.mme	Base64 — A format for encoding arbitrary binary information for transmission by electronic mail. [RFC3548]
Compress		Compress — UNIX compression [RFC1977].
DCS	.eps	Document Color Separation (DCS), version 2.0. [DCS2.0]
Deflate		Deflate — The file is compressed using ZIP public domain compression format [RFC1951].
GZip	.gz	Gzip — GNU zip compression technology [RFC1952].
MacBinary	.bin	MacBinary — An encoding format that combines the two forks of a Mac file, together with the file information (Name, Creator Application, File Type, etc.) into a single binary data stream that is suitable for storage or transferring through non-Mac systems. [macbinary]
Tar	.tar	UNIX packaging format.
TIFF/IT	.fp	TIFF/IT [ISO12639:2004] — Full Page — baseline Note: the file format TIFF/IT SHALL NOT use the “application/tiff” @MimeType. The “image/tiff” @MimeType conforms to baseline TIFF 6.0 [RFC3302], whereas TIFF/IT does not conform to TIFF 6.0. Consequently, the widely-deployed TIFF 6.0 readers are not able to read TIFF/IT. The [RFC3302] requires that an RFC be published in order to extend image/tiff with a parameter that would be needed in order to distinguish TIFF/IT from TIFF. There is no plan by the ISO committee that oversees TIFF/IT to register TIFF/IT with either a parameter to image/tiff or as new separate MIME type. Therefore, TIFF/IT will use the @FileType Attribute instead of the @MimeType Attribute.
TIFF/IT	.ct	TIFF/IT [ISO12639:2004] — Continuous Tone picture data — baseline
TIFF/IT	.lw	TIFF/IT [ISO12639:2004] — Continuous Line art — baseline

Table H-2: MimeType and File Type Combinations (Sheet 2 of 3)

MimeType	File Extension	Description [iana-mt] indicates IANA registration
TIFF/IT	.hc	TIFF/IT [ISO12639:2004] — High-resolution Continuous tone image data — baseline
TIFF/IT	.mp	TIFF/IT [ISO12639:2004] — monochrome picture image data — baseline
TIFF/IT	.bp	TIFF/IT [ISO12639:2004] — Binary Picture image data — baseline
TIFF/IT	.bl	TIFF/IT [ISO12639:2004] — Binary Line art image data — baseline
TIFF/IT	.fp	TIFF/IT [ISO12639:2004] — Full Page — profile 1
TIFF/IT	.ct	TIFF/IT [ISO12639:2004] — Continuous Tone picture data — profile 1
TIFF/IT	.lw	TIFF/IT [ISO12639:2004] — Color Line art data — profile 1
TIFF/IT	.hc	TIFF/IT [ISO12639:2004] — High-resolution Continuous tone image data — profile 1
TIFF/IT	.mp	TIFF/IT [ISO12639:2004] — monochrome picture image data — profile 1
TIFF/IT	.bp	TIFF/IT [ISO12639:2004] — Binary Picture image data — profile 1
TIFF/IT	.bl	TIFF/IT [ISO12639:2004] — Binary Line art image data — profile 1
TIFF/IT	.fp	TIFF/IT [ISO12639:2004] — Full Page — baseline Note: this entry and following ones were created in the context of [ISO12639:2004], whereas preceding entries were created in the context of the 1998 version of [ISO12639:2004]
TIFF/IT	.ct	TIFF/IT [ISO12639:2004] — Continuous Tone picture data — baseline
TIFF/IT	.lw	TIFF/IT [ISO12639:2004] — Color Line art data — baseline
TIFF/IT	.hc	TIFF/IT [ISO12639:2004] — High-resolution Continuous tone image data — baseline
TIFF/IT	.mp	TIFF/IT [ISO12639:2004] — monochrome picture image data — baseline
TIFF/IT	.bp	TIFF/IT [ISO12639:2004] — Binary Picture image data — baseline
TIFF/IT	.bl	TIFF/IT [ISO12639:2004] — Binary Line art image data — baseline
TIFF/IT	.sd	TIFF/IT [ISO12639:2004]
TIFF/IT	.fp	TIFF/IT [ISO12639:2004] — Full Page — profile 1
TIFF/IT	.ct	TIFF/IT [ISO12639:2004] — Continuous Tone picture data — profile 1
TIFF/IT	.lw	TIFF/IT [ISO12639:2004] — Color Line art data — profile 1
TIFF/IT	.hc	TIFF/IT [ISO12639:2004] — High-resolution Continuous tone image data — profile 1
TIFF/IT	.mp	TIFF/IT [ISO12639:2004] — monochrome picture image data — profile 1
TIFF/IT	.bp	TIFF/IT [ISO12639:2004] — Binary Picture image data — profile 1
TIFF/IT	.bl	TIFF/IT [ISO12639:2004] — Binary Line art image data — profile 1. Note: There is no TIFF/IT P1 conformance level of SD in [ISO12639:2004]
TIFF/IT	.fp	TIFF/IT [ISO12639:2004] — Full Page — profile 2
TIFF/IT	.ct	TIFF/IT [ISO12639:2004] — Continuous Tone picture data — profile 2
TIFF/IT	.lw	TIFF/IT [ISO12639:2004] — Color Line art data — profile 2
TIFF/IT	.hc	TIFF/IT [ISO12639:2004] — High-resolution Continuous tone image data — profile 2

Table H-2: MimeType and File Type Combinations (Sheet 3 of 3)

MimeType	File Extension	Description [iana-mt] indicates IANA registration
TIFF/IT	.mp	TIFF/IT [ISO12639:2004] — monochrome picture image data — profile 2
TIFF/IT	.bp	TIFF/IT [ISO12639:2004] — Binary Picture image data — profile 2
TIFF/IT	.bl	TIFF/IT [ISO12639:2004] — Binary Line art image data — profile 2
TIFF/IT	.sd	TIFF/IT [ISO12639:2004]
Type 1 Font	.pfa .pfb	Type 1 Font [type1font]
True Type Font	.ttf	True Type Font [truetypefont]
Open Type Font	.otf	Open Type Font [opentypefont]
UUEncoded	.uue	Uuencode — A set of encoding algorithms for converting files into a series of 7-bit ASCII characters that can be transmitted over the Internet. Originally, uuencode stood for Unix-to-Unix encode, but it has since become a universal protocol used to transfer files between different platforms such as Unix, Windows and Macintosh. Uuencoding is especially popular for sending Email attachments. [uuencode]
ZLIB		ZLIB — ZLIB compression [RFC1950]

Appendix HMimeType andMimeTypeVersion Attributes

Appendix I Generating strings with Format and Template

XJDF specifies a set of `@XXXFormat` `@XXXTemplate` pairs that allow dynamic generation of strings based on the standard C `printf()` function. (See [K&R].) The following instances are specified:

- `Subscription/@Format` and `Subscription/@Template`
- `FileSpec/@FileFormat` and `FileSpec/@FileTemplate`
- `IdentificationField/@ValueFormat` and `IdentificationField/@ValueTemplate`
- `JobField/@JobFormat` and `JobField/@JobTemplate`
- `JobField/@ShowList` with a fixed format
- `Layout/@SheetNameFormat` and `Layout/@SheetNameTemplate`
- `Layout/MarkObject/DynamicField@Format` and `Layout/MarkObject/DynamicField@Template`.
- `RunList/MetadataMap/@ValueFormat` and `RunList/MetadataMap/@ValueFormat`
- `StrippingParams/@SheetNameFormat` and `StrippingParams/@SheetNameTemplate`

The function defined when using the Attributes `@XXXFormat` and `@XXXTemplate` is based on the standard C `printf()` function. (See [K&R].) `@XXXFormat` is the first argument and `@XXXTemplate` is a comma-separated list of the additional arguments. `@XXXTemplate` MAY contain unary operators: “+” and “-”, binary operators: “+”, “-”, “*”, “/” and “%”, as well as parentheses: “(” and “)”, which are evaluated using standard C-operator precedence and the variables defined in the following table which include any valid Partition Key of a Partitioned Resource.

When evaluating a mathematical expression involving variables, the format evaluation will convert variable values as necessary into the long float values needed to result in a numeric result, and then convert that result if necessary based on what the format specifier is. For example, if:

- 1 A Template contains `"Metadata0 * Metadata1"`, and `@Metadata0 = "5"`, and `@Metadata1 = "1.5"`.
- 2 Both partition key values will be converted into long float values before the multiplication is performed
- 3 The result will be 7.5

Then, if the Format uses `%d`, the value will be truncated to 7. If the Format uses `%s`, the string `"7.5"` will be used. Finally, if the Format uses `%f`, the long float value 7.5 will be used.

If a mathematical operation is attempted on a non numeric value, the results are undefined.

Table I-1: Predefined variables used in @XXXTemplate and @ShowList (Sheet 1 of 3)

Name	Description
<code><a Partition Key></code>	Any Partition Key that is a value of <code>@PartIDKeys</code> in Table 8-1, “Resource Element” on page 517.
<code><any Imposition Variable></code>	The value of any variable defined by the Imposition process (see Section 6.4.18.2, “Variables for Automated Imposition”).
<code>all</code>	Selects all matching Elements. Valid only when FileSpec is used as an Input Resource.
<code>Amount</code>	Amount of the product that was produced.
<code>CustomerID</code>	<code>@CustomerID</code> .
<code>Date</code>	Current <code>@Date</code> in [ISO8601:2004] format.

Table I-1: Predefined variables used in @XXXTemplate and @ShowList (Sheet 2 of 3)

Name	Description
<i>DeviceID</i>	ID of the Device. This is a unique name within the workflow.
<i>element</i>	Integer iterator over all elements in a given page. Restarts at 0 for each page.
<i>EndTime</i>	Actual end time of the Job.
<i>Error</i>	Errors that happened during the Job.
<i>ErrorStats</i>	Statistics on errors that happened during execution.
<i>ExposedMediaName</i>	@DescriptiveName of the exposed media (e.g., plate or proof that is being imaged).
<i>FriendlyName</i>	@FriendlyName of the Device.
<i>GeneralID:XXX</i>	GeneralID/@IDValue of a GeneralID[@IDUsage = "XXX"] For example if @Format = "%i" and @Template= "GeneralID:foo" then for: <code><GeneralID IDUsage="foo" IDValue="1"/></code> the extracted value is 1.
<i>Generated</i>	System generated file name.
<i>i</i>	Integer iterator over all files produced by this Process. 0-based numbering.
<i>input</i>	Local file name of the input file. Valid only when FileSpec is used as an Output Resource.
<i>jobID</i>	Job ID .
<i>jobName</i>	@DescriptiveName of the Node that is being processed.
<i>jobPartID</i>	@JobPartID string.
<i>JobRecipientName</i>	Name of the recipient of the Job.
<i>JobSubmitterName</i>	Name of the submitter of the Job.
<i>MediaBrand</i>	Brand of the media that is being printed.
<i>MediaType</i>	"DescriptiveName" of the media that is being printed.
<i>MoonPhase</i>	Phase of the moon at the @StartTime of the Job.
<i>Operator</i>	Name of the operator.
<i>OperatorText</i>	Text from the operator as defined in @OperatorText.
<i>page</i>	Integer iterator over the page number of a document. This value is equivalent to value <i>r</i> (below) for the case that each run contains exactly one page.
<i>PrintQuality</i>	The quality of the printout. (High, Normal, Draft or Device specific name)
<i>ProoferProfileName</i>	Name of the ICC profile for the proofing Device.
<i>PressProfileName</i>	Name of the ICC profile for the final printing (used as intermediate space during proofing).
<i>r</i>	Integer iterator over all RunList Partitions with a Partition Key of "Run" in an input RunList .
<i>ri</i>	Integer iterator over all indices in an input "Run" of a RunList . This index is equivalent to looping over a @RunIndex.
<i>Resolution</i>	Output resolution.
<i>ResolutionX</i>	Output resolution in X direction.
<i>ResolutionY</i>	Output resolution in Y direction.

Table I-1: Predefined variables used in @XXXTemplate and @ShowList (Sheet 3 of 3)

Name	Description
<i>ScreeningFamily</i>	Name of the screening family of the output.
<i>SheetNum</i>	Integer iterator over the sheet number of a document.
<i>StartTime</i>	Actual start time of the Job
<i>SystemRoot</i>	Root of system directory file structure. This token provides an operating system, independent way to refer to the root.
<i>TileX</i>	X coordinate of a Tile.
<i>TileY</i>	Y coordinate of a Tile.
<i>Time</i>	Current <i>@Time</i> in [ISO8601:2004] format.
<i>TotalPagesInDoc</i>	Total # of pages in a document.
<i>UserText</i>	For JobField, " <i>UserText</i> " references user-defined text in JobField/ <i>@UserText</i> .
<i>Warning</i>	Warnings that happened during the Job. Warnings don't lose information in the resulting Job, while errors do.

Example I-1: @FileTemplate and @FileFormat

With *@JobID="j001"* and a **RunList** defining 2024 created files, this example will iterate over all created files and place them into:

```
"file://myserver.mydomain.com/next/j001/0000/m0000.pdf"
...
"file://myserver.mydomain.com/next/j001/0020/m0023.pdf"
```

TBD 2.x Example.

```
<RunList Class="Parameter" ID="R1" Status="Available">
  <LayoutElement>
    <FileSpec FileFormat=
      "file://myserver.mydomain.com/next/%s/%4.i/m%4.i.pdf"
      FileTemplate="JobID,i/100,i%100"/>
  </LayoutElement>
</RunList>
```


Appendix J Pagination Catalog

This appendix provides a set of diagrams that explain how pages are arranged in groups when preparing to print on the surfaces of large sheets. The diagrams show a wide range of folding patterns to be used before binding. The folding patterns are specified in the **XJDF** Fold Catalog (see Section 8-34, “Fold catalog part 1” on page 674 and Section 8-35, “Fold catalog part 2” on page 675) which describes how to paginate single-sheet bindery signatures.

The purpose of this appendix is to provide a reference for all agents involved in the use of imposition techniques in the printing industry.

J.1 How to interpret the diagrams

J.1.1 Legend

This document describes the structure and arrangement of bindery signatures into pagination schemes, which divide sheet surfaces into grids of rectangular areas to be filled by document pages during imposition process. These arrangements are the consequence of manipulations made on the sheets by folding, trimming and binding them in order to make booklets ready for assembly.

This document uses diagrams to describe the pagination schemes. Each diagram shows a side of an unfolded sheet, illustrating how it is divided into "signature cells". All cells are usually of the same size, allowing the entire sheet to be divided into equal portions, with each portion covering the whole area between surrounding folds. A signature cell is the space that "receives" a single document page and surrounding margins that are part of the gutters.

Each cell shown in the diagram displays how to orient the document page that is to be imposed there, and specifies the index of the page to be imposed. This means that the resulting booklet will have pages that are properly ordered and properly oriented in the product reader view, according to the default values defined in the this [XJDF] specification. Note that pagination indexes start at number 1.

The diagrams also show the pagination to be used when pages are flowed in reverse order because of different binding options (see sections Section J.1.3, “Settings that Modify the Pagination Schemes” and Section J.1.4, “Getting a Specific Pagination Scheme”).

Folding sequences are described using the same notation found in this [XJDF] Specification:

- \uparrow means "left-hand part goes over right-hand part" or "bottom part goes over top part";
- \downarrow means "left-hand part goes under right-hand part" or "bottom part goes under top part";
- The size of the part being folded is expressed as a fraction of unfolded sheet's size;
- First fold is always left-to-right, a "+" sign is used to toggle between left-to-right and bottom-to-top folding directions.

J.1.2 Meaning of a Pagination Scheme

The diagrams in Section J.2, “Pagination Diagrams” show the configuration of the page cells that occurs when the bindery signature is specified in the **XJDF** file using the **BinderySignature/@FoldCatalog** attribute. Each arrangement corresponds to the "fold catalog identifier" that is shown in the left column of the diagram. This identifier is used as a value of the **BinderySignature/@FoldCatalog** attribute, and refers to the recipe used to fold the sheet.

The pagination indexes shown in the diagram correspond to the imposition order, starting with 1, up to the number of pages in a booklet. This index does not correspond to the actual page numbers that will be imposed on the sheets, unless a finished product is made of a single booklet and the first page is numbered “1”. These numbers specify the order that pages are imposed into signature cells, from an array of pages associated to a booklet.

These numbers have meaning only if the folded sheet are used as a booklet intended for binding or assembling (i.e., trimmed after folding). In many cases (i.e., accordion folds) the result is mostly theoretical, as those folds are not intended for such use in real life.

When multiple booklets are assembled together, the imposition indexes have to be translated into numbers referring to the list of source document pages. This is calculated using the parameters found in the Assembly resource and the **StrippingParams/@SectionList** parameter.

The numbers and page orientation shown in the diagram correspond to the finished product view in reader's perspective. The "top of the page", which is a product attribute, does not always correspond to the "head" of the booklet, which is a production attribute. Note, that the finished view is NOT a reference **for locating the production measurements** (head/foot/face trim and bleed sizes, spine size, overfolds, etc.) as their position is set by the **@BindingOrientation** attribute, independently of the final page flow.

J.1.3 Settings that Modify the Pagination Schemes

J.1.3.1 BindingOrientation

When a sheet has been folded, the last fold is recognized as being the "binding edge", and a perpendicular edge is known as the "jog edge". Both edges join together around a corner known as the "reference corner", which appears at the bottom left of the folded sheet (the last fold always appear either at the bottom or at the left when using the fold catalog).

The attribute **BinderySignature/@BindingOrientation** may be set to indicate that the reference corner is displaced for production purposes. This manipulation is not made on the folded sheet. Only the "virtual" corner is changed, after edges had been identified. This means that the edges that are recognized as "binding" and "jog" are found at new places on the folded sheet, changing the location of the spine, head, face and foot on the booklet before pagination can be applied.

This **@BindingOrientation** attribute is very special because it has two default values, depending on the type of signature being defined: "*Flip0*" for single-row grids (no closed head), "*Rotate0*" for all other grids. This particularity reflects common practice of recognizing the jog edge to be at the top of signatures without closed heads.

The diagrams in Section J.2, "Pagination Diagrams" are based on these default values. If that parameter is set to another value, use the tables in Section J.1.4, "Getting a Specific Pagination Scheme" to convert the pagination scheme to reflect this change.

J.1.3.2 Binding and Jog Sides

To make the bindery signatures be assembled together into a finished product, the pages must have been imposed in the order and orientation needed to get the right reader's perspective after assembling. Before setting the page numbers and orientation in cells to obtain the expected result, the "assembly" is virtually rotated and flipped to make the binding and jog edges be placed as requested, when looking at the very first page in the reader view.

The diagrams in this document show the pagination scheme where the front page of the booklet is oriented so that the binding side appears at the left, and the jog side appears at the top, according to the default parameters defined by this [XJDF] specification. For other values, transformations must be applied on the diagrams to get the right scheme.

These settings are found in the Assembly resource that is used to describe how the booklet is assembled:

- **Assembly/@BindingSide**
- **Assembly/@JogSide**

If one or both of these attributes is set, use the tables in Section J.1.4, "Getting a Specific Pagination Scheme" to convert the pagination scheme to reflect this change.

The settings **BinderySignature/@BindingEdge** and **BinderySignature /@JogEdge** are ignored because they affect production view only. However, if **Assembly/@Order="None"**, then **BinderySignature/**

`@BindingEdge` and `BinderySignature/@JogEdge` must be used as replacement settings, because assembly parameters must be ignored in that case, and production view becomes the product view.

J.1.4 Getting a Specific Pagination Scheme

J.1.4.1 Using the Settings to Find the Needed Scheme Transformation

Use the table below to locate the name of the scheme transformation to be applied on the diagram, according to the `@BindingSide`, `@JogSide` and `@BindingOrientation` settings. Default values for these settings are underlined in the table.

The obtained transformation is identified by a "scheme name", which refers to the table in Section J.1.4.2, "Scheme Transformations" where all pagination schemes are explained, based on the diagrams of the chapter 2.

Table J-1: Schemes Names for Binding Orientations

		Scheme Name (for <code>@BindingOrientation</code> setting shown in header)				
		(for single-row signatures: use column footers)				
<code>@Binding Side</code>	<code>@JogSide</code>	<u>Rotate0</u> <u>Flip0</u>	<u>Rotate90</u> <u>Flip90</u>	<u>Rotate180</u> <u>Flip180</u>	<u>Rotate270</u> <u>Flip270</u>	
		top left	Rotate0 <u>Flip0</u>	Rotate270/90* <u>Flip90/270*</u>	Rotate180 <u>Flip180</u>	Rotate90/270* <u>Flip270/90*</u>
		bottom	Flip0 <u>Rotate0</u>	Flip270/90* <u>Rotate90/270*</u>	Flip180 <u>Rotate180</u>	Flip90/270* <u>Rotate270/90*</u>
right		top	Flip180 <u>Rotate180</u>	Flip90/270* <u>Rotate270/90*</u>	Flip0 <u>Rotate0</u>	Flip270/90* <u>Rotate90/270*</u>
		bottom	Rotate180 <u>Flip180</u>	Rotate90/270* <u>Flip270/90*</u>	Rotate0 <u>Flip0</u>	Rotate270/90* <u>Flip90/270*</u>
top		left	Flip90 <u>Rotate90</u>	Flip180/0 <u>Rotate180/0</u>	Flip270 <u>Rotate270</u>	Flip0/180 <u>Rotate0/180</u>
		right	Rotate90 <u>Flip90</u>	Rotate0/180 <u>Flip0/180</u>	Rotate270 <u>Flip270</u>	Rotate180/0 <u>Flip180/0</u>
bottom		left	Rotate270 <u>Flip270</u>	Rotate180/0 <u>Flip180/0</u>	Rotate90 <u>Flip90</u>	Rotate0/180 <u>Flip0/180</u>
		right	Flip270 <u>Rotate270</u>	Flip0/180 <u>Rotate0/180</u>	Flip90 <u>Rotate90</u>	Flip180/0 <u>Rotate180/0</u>
		<u>Flip0</u> <u>Rotate0</u>	<u>Flip270</u> <u>Rotate270</u>	<u>Flip180</u> <u>Rotate180</u>	<u>Flip90</u> <u>Rotate90</u>	

* **Important note:** if binding edges appear horizontally on the diagram, the numbers must be swapped in the scheme names indicated by an asterisk ("Rotate90/270" would become "Rotate270/90"). This happens because the direction of rotation is reversed in those cases (e.g., F8-7, F12-7, F16-10, etc.).

J.1.4.2 Scheme Transformations

Table J-2: Transformations for each Scheme (Sheet 1 of 2)

Scheme Name	Getting the pagination scheme (using the diagram)		
	Page Numbers	Left-Bound Page	Right-Bound Page
Rotate0	Normal	as shown	as shown

Table J-2: Transformations for each Scheme (Sheet 2 of 2)

Scheme Name	Getting the pagination scheme (using the diagram)		
	Page Numbers	Left-Bound Page	Right-Bound Page
Rotate0/180	Normal	as shown	Rotate 180°
Rotate90	Normal	Rotate 90° counterclockwise	Rotate 90° counterclockwise
Rotate90/270	Normal	Rotate 90° counterclockwise	Rotate 90° clockwise
Rotate180	Normal	Rotate 180°	Rotate 180°
Rotate180/0	Normal	Rotate 180°	as shown
Rotate270	Normal	Rotate 90° clockwise	Rotate 90° clockwise
Rotate270/90	Normal	Rotate 90° clockwise	Rotate 90° counterclockwise
Flip0	Reverse	Rotate 180°	Rotate 180°
Flip0/180	Reverse	Rotate 180°	as shown
Flip90	Reverse	Rotate 90° clockwise	Rotate 90° clockwise
Flip90/270	Reverse	Rotate 90° clockwise	Rotate 90° counterclockwise
Flip180	Reverse	as shown	as shown
Flip180/0	Reverse	as shown	Rotate 180°
Flip270	Reverse	Rotate 90° counterclockwise	Rotate 90° counterclockwise
Flip270/90	Reverse	Rotate 90° counterclockwise	Rotate 90° clockwise

In the “Page Numbers” column in the above table, the value “Normal” refers to the main numbers in Table J-3, Table J-5 and Table J-7, while “Reverse” refers to the smaller numbers in gray. Note that the small gray numbers have been omitted from Table J-4 and Table J-6. If they were shown, they would be the same as for Table J-3 and Table J-5, respectively.

The “Left-Bound Pages” column in the above table refers to odd pages in the tables below, when looking at the main numbers. The “Right-Bound Pages” column in the above table refers to even pages.

Important note: when a page is rotated 90° (clockwise or counterclockwise), this rotation is made inside the signature cell. The cell itself is not rotated because the folding operation remains the same. This means that the aspect ratio of the page must have been designed accordingly. Observe this situation in the examples in the next section.

J.1.5 Examples

J.1.5.1 Signature with Horizontal Binding Edges

The examples below show how to read the diagrams after applying the transformations explained previously. Each diagram is an interpretation of the lay-side diagram defined for fold catalog F8-7, indicating the scheme name above it

Table J-3: Original Diagram

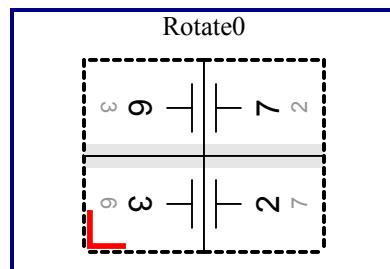


Table J-4: Horizontal Binding Edges

Rotate0	Rotate90	Rotate180	Rotate270
A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.
Rotate0/180	Rotate90/270	Rotate180/0	Rotate270/90
A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.
Flip0	Flip90	Flip180	Flip270
A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.
Flip0/180	Flip90/270	Flip180/0	Flip270/90
A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.	A 4x4 grid with dashed lines forming a 3x3 inner square. The numbers 9, 6, 3, 2, 7 are arranged in a 2x2 pattern within the inner square. A red L-shaped arrow at the bottom-left corner indicates a horizontal binding edge.

J.1.5.2 Signature with Vertical Binding Edges

The examples below show how to read the diagrams after applying the transformations explained previously. Each diagram is an interpretation of the lay-side diagram defined for fold catalog F12-11, indicating the scheme name above it

Table J-5: Original Diagram

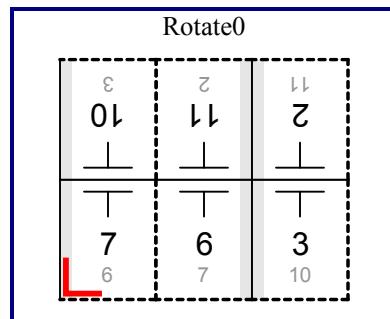


Table J-6: Vertical Binding Edges

Rotate0	Rotate90	Rotate180	Rotate270
A 4x3 grid of squares. The squares are labeled with numbers: 10, 11, 2, 11, 10, 2 in the top row; 7, 6, 3, 10, 7, 6 in the second row; 9, 3, 2, 3, 9, 7 in the third row; and 6, 7, 11, 10, 11, 7 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 10, 11, 2, 11, 10, 2 in the top row; 7, 6, 3, 10, 7, 6 in the second row; 9, 3, 2, 3, 9, 7 in the third row; and 6, 7, 11, 10, 11, 7 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 10, 11, 2, 11, 10, 2 in the top row; 7, 6, 3, 10, 7, 6 in the second row; 9, 3, 2, 3, 9, 7 in the third row; and 6, 7, 11, 10, 11, 7 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 10, 11, 2, 11, 10, 2 in the top row; 7, 6, 3, 10, 7, 6 in the second row; 9, 3, 2, 3, 9, 7 in the third row; and 6, 7, 11, 10, 11, 7 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.
Rotate0/180	Rotate90/270	Rotate180/0	Rotate270/90
A 4x3 grid of squares. The squares are labeled with numbers: 10, 11, 2, 11, 10, 2 in the top row; 7, 6, 3, 10, 7, 6 in the second row; 9, 3, 2, 3, 9, 7 in the third row; and 6, 7, 11, 10, 11, 7 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 10, 11, 2, 11, 10, 2 in the top row; 7, 6, 3, 10, 7, 6 in the second row; 9, 3, 2, 3, 9, 7 in the third row; and 6, 7, 11, 10, 11, 7 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 10, 11, 2, 11, 10, 2 in the top row; 7, 6, 3, 10, 7, 6 in the second row; 9, 3, 2, 3, 9, 7 in the third row; and 6, 7, 11, 10, 11, 7 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 10, 11, 2, 11, 10, 2 in the top row; 7, 6, 3, 10, 7, 6 in the second row; 9, 3, 2, 3, 9, 7 in the third row; and 6, 7, 11, 10, 11, 7 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.
Flip0	Flip90	Flip180	Flip270
A 4x3 grid of squares. The squares are labeled with numbers: 3, 2, 11, 10, 7, 6 in the top row; 9, 6, 7, 10, 11, 2 in the second row; 6, 7, 11, 10, 11, 2 in the third row; and 3, 2, 11, 10, 7, 6 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 3, 2, 11, 10, 7, 6 in the top row; 9, 6, 7, 10, 11, 2 in the second row; 6, 7, 11, 10, 11, 2 in the third row; and 3, 2, 11, 10, 7, 6 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 3, 2, 11, 10, 7, 6 in the top row; 9, 6, 7, 10, 11, 2 in the second row; 6, 7, 11, 10, 11, 2 in the third row; and 3, 2, 11, 10, 7, 6 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 3, 2, 11, 10, 7, 6 in the top row; 9, 6, 7, 10, 11, 2 in the second row; 6, 7, 11, 10, 11, 2 in the third row; and 3, 2, 11, 10, 7, 6 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.
Flip0/180	Flip90/270	Flip180/0	Flip270/90
A 4x3 grid of squares. The squares are labeled with numbers: 3, 2, 11, 10, 7, 6 in the top row; 9, 6, 7, 10, 11, 2 in the second row; 6, 7, 11, 10, 11, 2 in the third row; and 3, 2, 11, 10, 7, 6 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 3, 2, 11, 10, 7, 6 in the top row; 9, 6, 7, 10, 11, 2 in the second row; 6, 7, 11, 10, 11, 2 in the third row; and 3, 2, 11, 10, 7, 6 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 3, 2, 11, 10, 7, 6 in the top row; 9, 6, 7, 10, 11, 2 in the second row; 6, 7, 11, 10, 11, 2 in the third row; and 3, 2, 11, 10, 7, 6 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.	A 4x3 grid of squares. The squares are labeled with numbers: 3, 2, 11, 10, 7, 6 in the top row; 9, 6, 7, 10, 11, 2 in the second row; 6, 7, 11, 10, 11, 2 in the third row; and 3, 2, 11, 10, 7, 6 in the bottom row. A red L-shaped arrow at the bottom-left corner indicates a vertical binding edge.

J.2 Pagination Diagrams

Table J-7: Pagination Diagrams (Sheet 1 of 41)

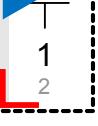
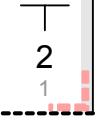
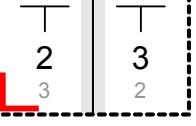
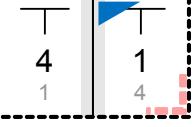
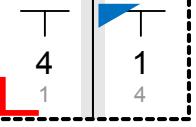
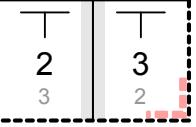
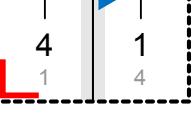
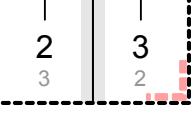
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F2-1	1x1		
	(no fold)		
F4-1	2x1		
	$\uparrow^{1/2}$		
F4-2	2x1		
	$\downarrow^{1/2}$		
F4-2	2x1		
	$\downarrow^{1/2}$		

Table J-7: Pagination Diagrams (Sheet 2 of 41)

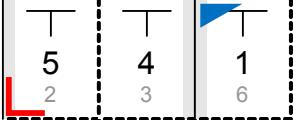
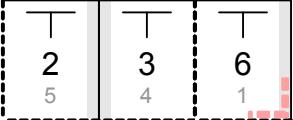
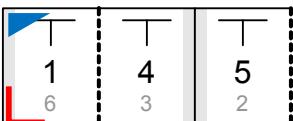
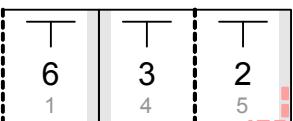
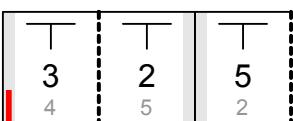
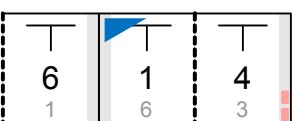
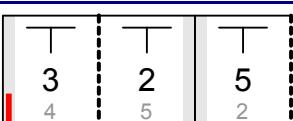
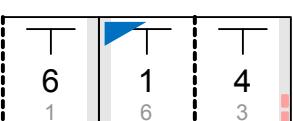
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F6-1	3x1		
	$\uparrow^{1/3} \downarrow^{1/3}$		
F6-2	3x1		
	$\downarrow^{1/3} \uparrow^{1/3}$		
F6-3	3x1		
	$\uparrow^{1/4} \uparrow^{1/2}$		
F6-4	3x1		
	$\uparrow^{1/3} \uparrow^{1/3}$		

Table J-7: Pagination Diagrams (Sheet 3 of 41)

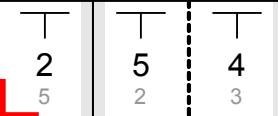
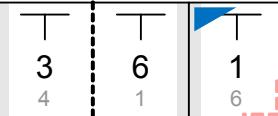
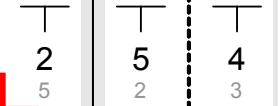
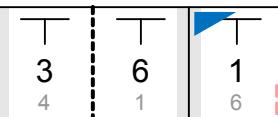
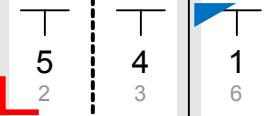
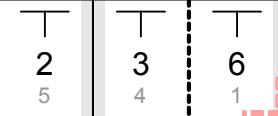
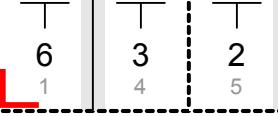
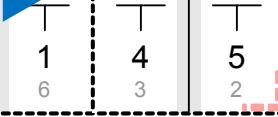
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F6-5	3x1		
		$\uparrow^2/3 \downarrow^{1/3}$	
F6-6	3x1		
		$\uparrow^{3/4} \downarrow^{1/4}$	
F6-7	3x1		
		$\uparrow^{1/4} \downarrow^{1/4}$	
F6-8	3x1		
		$\uparrow^{2/3} \uparrow^{1/3}$	

Table J-7: Pagination Diagrams (Sheet 4 of 41)

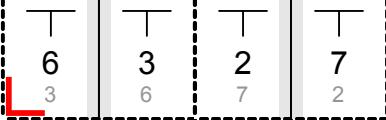
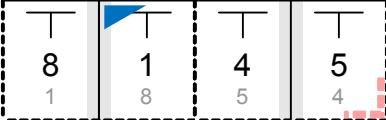
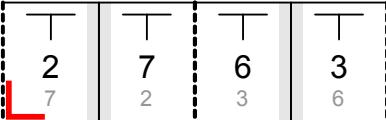
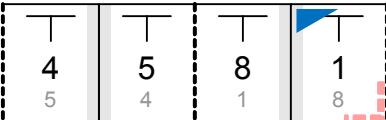
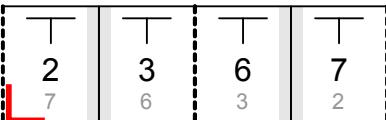
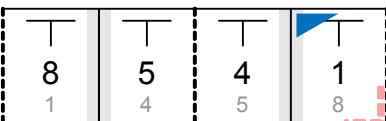
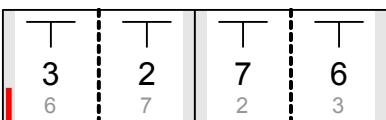
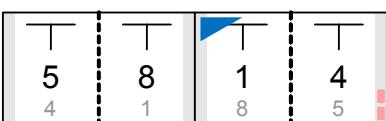
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F8-1	4x1		
	$\uparrow^{1/2} \uparrow^{1/4}$		
F8-2	4x1		
	$\uparrow^{1/2} \downarrow^{1/4}$		
F8-3	4x1		
	$\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4}$		
F8-4	4x1		
	$\uparrow^{1/4} \uparrow^{1/2} \downarrow^{1/4}$		

Table J-7: Pagination Diagrams (Sheet 5 of 41)

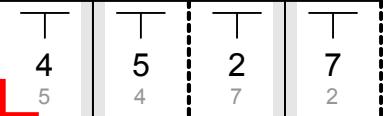
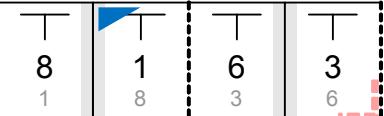
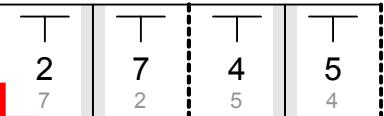
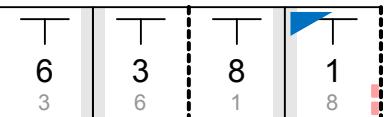
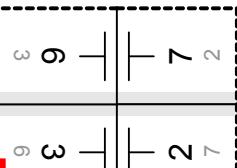
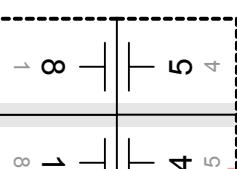
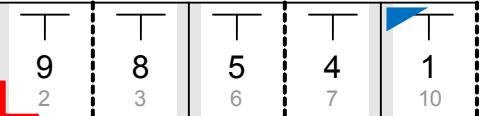
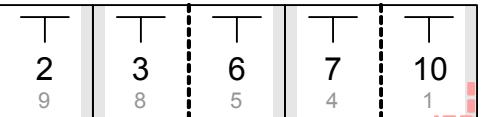
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F8-5	4x1		
	$\uparrow^{1/4} \uparrow^{1/4} \uparrow^{1/4}$		
F8-6	4x1		
	$\uparrow^{3/4} \downarrow^{1/4} \downarrow^{1/4}$		
F8-7	2x2	 	
	$\uparrow^{1/2} + \uparrow^{1/2}$		
F10-1	5x1		
	$\uparrow^{1/5} \downarrow^{1/5} \uparrow^{1/5} \downarrow^{1/5}$		

Table J-7: Pagination Diagrams (Sheet 6 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)					Back Side (Non-lay Side)						
Folding Sequence													
F10-2	5x1	 Front side diagram for F10-2 showing a 5x1 grid with fold lines and page numbers 2, 9, 4, 7, 6. A red L-shaped fold indicator is at the bottom-left corner.					 Back side diagram for F10-2 showing a 5x1 grid with fold lines and page numbers 5, 8, 3, 10, 1. A blue T-shaped fold indicator is at the top-right corner.						
		$\uparrow^4/5 \downarrow^1/5 \downarrow^1/5 \downarrow^1/5$											
F10-3	5x1	 Front side diagram for F10-3 showing a 5x1 grid with fold lines and page numbers 2, 9, 8, 3, 6. A red L-shaped fold indicator is at the bottom-left corner.					 Back side diagram for F10-3 showing a 5x1 grid with fold lines and page numbers 5, 4, 7, 10, 1. A blue T-shaped fold indicator is at the top-right corner.						
		$\uparrow^2/5 \downarrow^2/5 \uparrow^1/5$											
F12-1	6x1	 Front side diagram for F12-1 showing a 6x1 grid with fold lines and page numbers 2, 11, 10, 3, 6, 7. A red L-shaped fold indicator is at the bottom-left corner.						 Back side diagram for F12-1 showing a 6x1 grid with fold lines and page numbers 8, 5, 4, 9, 12, 1. A blue T-shaped fold indicator is at the top-right corner.					
		$\uparrow^1/3 \downarrow^1/3 \uparrow^1/6$											
F12-2	6x1	 Front side diagram for F12-2 showing a 6x1 grid with fold lines and page numbers 10, 3, 2, 11, 8, 5. A red L-shaped fold indicator is at the bottom-left corner.						 Back side diagram for F12-2 showing a 6x1 grid with fold lines and page numbers 6, 7, 12, 1, 4, 9. A blue T-shaped fold indicator is at the top-right corner.					
		$\uparrow^1/3 \uparrow^1/3 \downarrow^1/6$											

Table J-7: Pagination Diagrams (Sheet 7 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)										
		Back Side (Non-lay Side)										
Folding Sequence												
F12-3	6x1											
F12-4	6x1											
F12-5	6x1											
F12-6	6x1											

Table J-7: Pagination Diagrams (Sheet 8 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F12-7	3x2		
		$\uparrow^{1/3} \downarrow^{1/3} + \uparrow^{1/2}$	
F12-8	3x2		
		$\uparrow^{2/3} \uparrow^{1/3} + \uparrow^{1/2}$	
F12-9	3x2		
		$\uparrow^{1/3} \uparrow^{1/3} + \uparrow^{1/2}$	

Table J-7: Pagination Diagrams (Sheet 9 of 41)

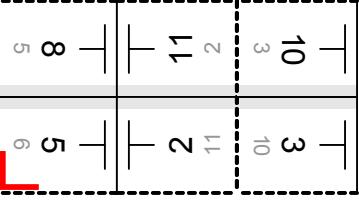
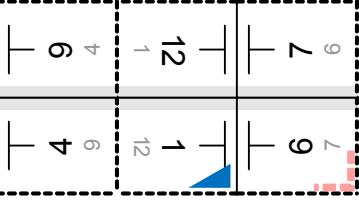
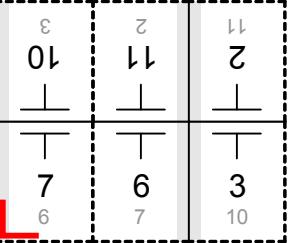
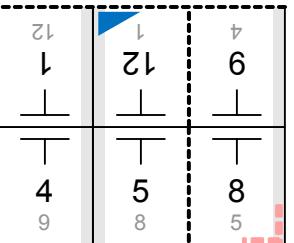
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	
		Folding Sequence	Back Side (Non-lay Side)
F12-10	3x2		
		$\uparrow^2/3 \downarrow^1/3 + \uparrow^1/2$	
F12-11	3x2		
		$\uparrow^1/3 + \uparrow^1/2 + \uparrow^1/3$	

Table J-7: Pagination Diagrams (Sheet 10 of 41)

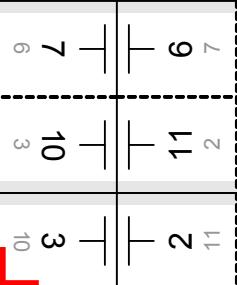
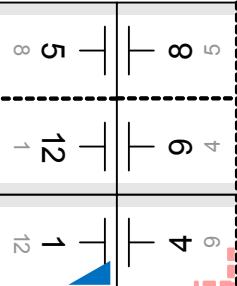
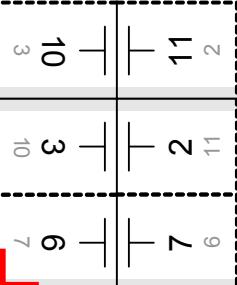
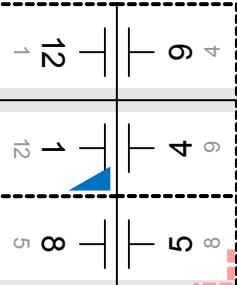
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F12-12	2x3		
F12-13	2x3		

Table J-7: Pagination Diagrams (Sheet 11 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F12-14	2x3		
		$\uparrow^{1/2} + \uparrow^{1/3} \downarrow^{1/3}$	
F14-1	7x1		
		$\uparrow^{1/7} \downarrow^{1/7} \uparrow^{1/7}$ $\downarrow^{1/7} \uparrow^{1/7} \downarrow^{1/7}$	
F16-1	8x1		
		$\uparrow^{1/2} \downarrow^{1/4} \uparrow^{1/8}$	

Table J-7: Pagination Diagrams (Sheet 12 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)							
Folding Sequence		Back Side (Non-lay Side)							
F16-2	8x1								
$\uparrow^{1/2} \downarrow^{1/4} \downarrow^{1/8}$									
F16-3	8x1								
$\uparrow^{1/2} \uparrow^{1/4} \downarrow^{1/8}$									
F16-4	8x1								
$\uparrow^{1/2} \uparrow^{1/4} \uparrow^{1/8}$									
F16-5	8x1								
$\downarrow^{1/8} \uparrow^{1/8} \downarrow^{1/8} \uparrow^{1/8}$ $\downarrow^{1/8} \uparrow^{1/8} \downarrow^{1/8}$									

Table J-7: Pagination Diagrams (Sheet 13 of 41)

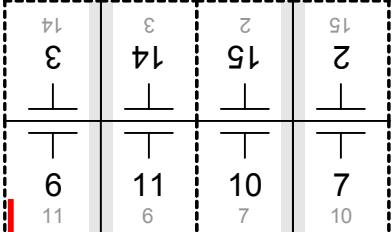
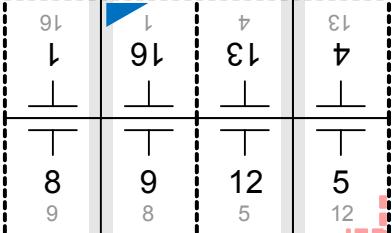
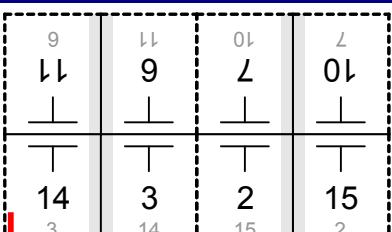
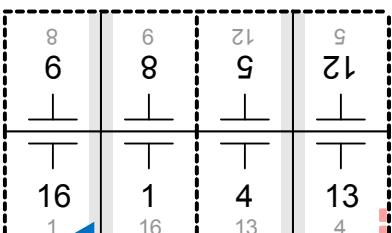
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F16-6	4x2		
			$\uparrow^1/2 + \uparrow^1/2 + \uparrow^1/4$
F16-7	4x2		
			$\uparrow^1/2 + \uparrow^1/2 + \downarrow^1/4$

Table J-7: Pagination Diagrams (Sheet 14 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F16-8	4x2		
	$\uparrow^{1/2} + \downarrow^{1/2} + \downarrow^{1/4}$		
F16-9	4x2		
	$\uparrow^{1/2} \downarrow^{1/4} + \uparrow^{1/2}$		

Table J-7: Pagination Diagrams (Sheet 15 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	
		Folding Sequence	Back Side (Non-lay Side)
F16-10	4x2		
		$\uparrow^{1/2} \uparrow^{1/4} + \uparrow^{1/2}$	
F16-11	4x2		
		$\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4} + \uparrow^{1/2}$	
F16-12	4x2		
		$\uparrow^{1/4} \uparrow^{1/4} \uparrow^{1/4} + \uparrow^{1/2}$	

Table J-7: Pagination Diagrams (Sheet 16 of 41)

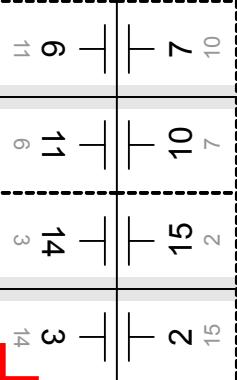
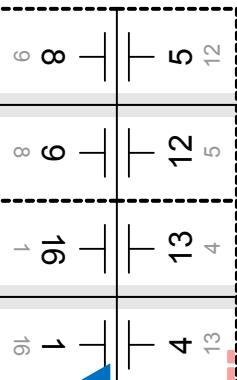
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F16-13	2x4		
	$\uparrow^1/2 + \uparrow^1/2 \downarrow^1/4$		

Table J-7: Pagination Diagrams (Sheet 17 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F16-14	2x4		
F18-1	9x1		
		$\uparrow^1/2 + \uparrow^1/2 \uparrow^1/4$	$\uparrow^1/9 \downarrow^1/9 \uparrow^1/9 \downarrow^1/9$ $\uparrow^1/9 \downarrow^1/9 \uparrow^1/9 \downarrow^1/9$

Table J-7: Pagination Diagrams (Sheet 18 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)								
		Back Side (Non-lay Side)								
Folding Sequence										
F18-2	9x1									
$\uparrow^2/3 \downarrow^1/3 \uparrow^1/9 \downarrow^1/9$										
F18-3	9x1									
$\uparrow^1/3 \downarrow^1/3 \uparrow^2/9 \downarrow^1/9$										
F18-4	9x1									
$\uparrow^1/3 \downarrow^1/3 \uparrow^1/9 \downarrow^1/9$										

Table J-7: Pagination Diagrams (Sheet 19 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	
		Folding Sequence	Back Side (Non-lay Side)
F18-5	3x3		
		$\uparrow 1/3 \downarrow 1/3 + \uparrow 1/3 \downarrow 1/3$	
F18-6	3x3		
		$\uparrow 1/3 \downarrow 1/3 + \uparrow^2 1/3 \downarrow 1/3$	

Table J-7: Pagination Diagrams (Sheet 20 of 41)

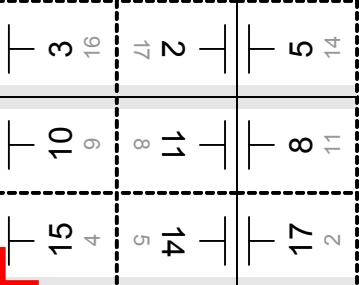
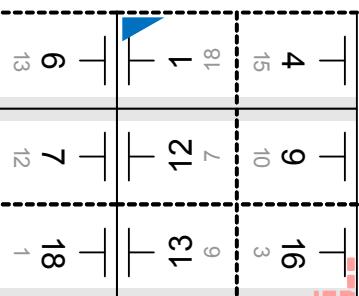
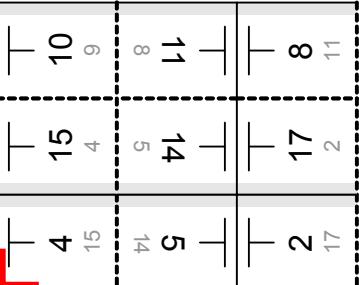
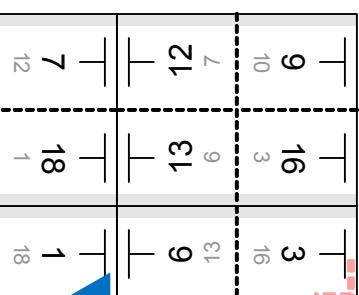
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F18-7	3x3		
		$\uparrow^{1/3} \uparrow^{1/3} + \uparrow^{1/3} \downarrow^{1/3}$	
F18-8	3x3		
		$\uparrow^{1/3} \uparrow^{1/3} + \uparrow^{2/3} \downarrow^{1/3}$	

Table J-7: Pagination Diagrams (Sheet 21 of 41)

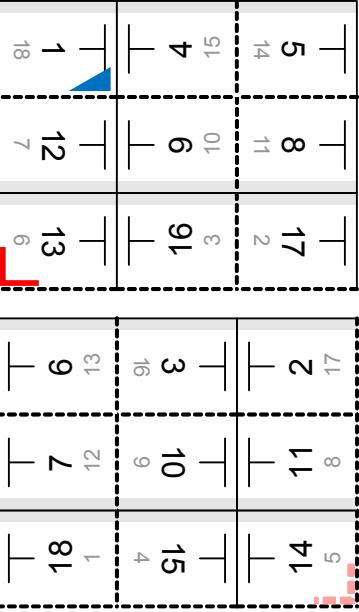
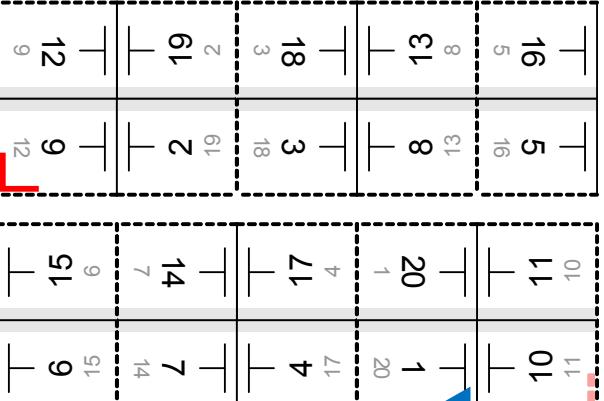
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F18-9	3x3		
		$\uparrow^{2/3} \downarrow^{1/3} + \uparrow^{2/3} \uparrow^{1/3}$	
F20-1	5x2		
		$\uparrow^{2/5} \downarrow^{2/5} \uparrow^{1/5} + \uparrow^{1/2}$	

Table J-7: Pagination Diagrams (Sheet 22 of 41)

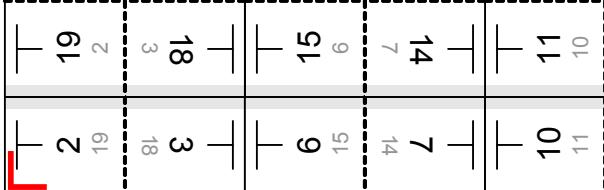
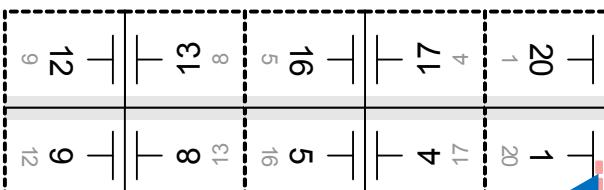
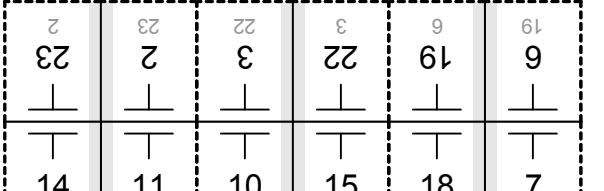
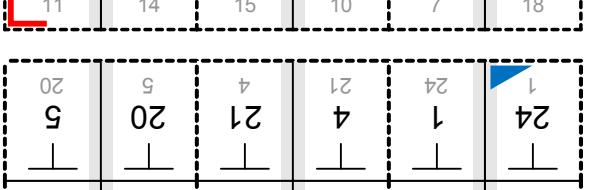
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	Back Side (Non-lay Side)
Folding Sequence			
F20-2	5x2		
		$\uparrow^1 /_5 \downarrow^1 /_5 \uparrow^1 /_5 \downarrow^1 /_5 + \uparrow^1 /_2$	
F24-1	6x2		
		$\uparrow^1 /_3 \downarrow^1 /_3 + \uparrow^1 /_6 + \uparrow^1 /_2$	

Table J-7: Pagination Diagrams (Sheet 23 of 41)

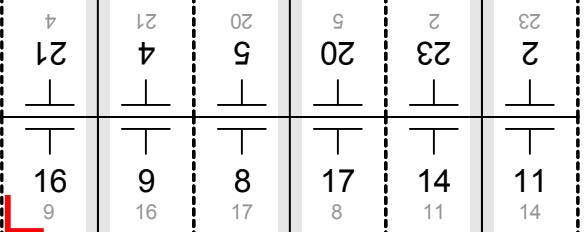
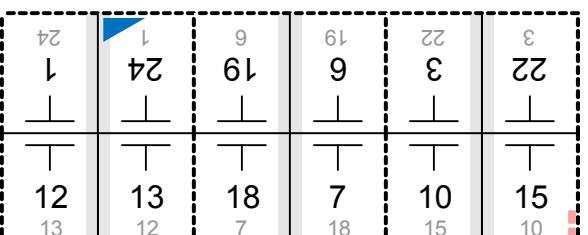
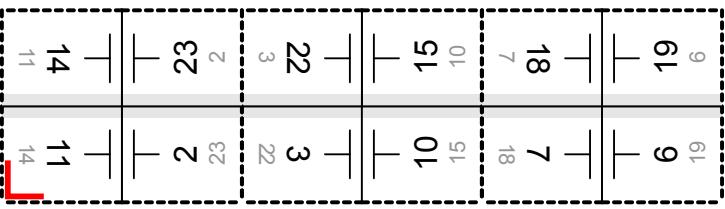
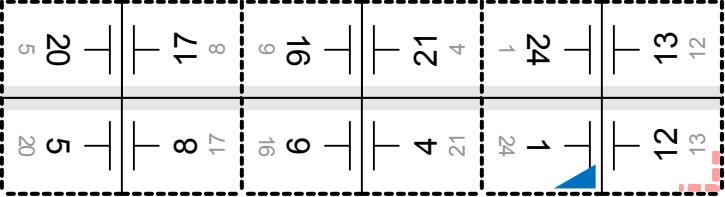
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	
Folding Sequence		Back Side (Non-lay Side)	
F24-2	9x2		 
F24-3	6x2	$\uparrow 1/3 \uparrow 1/3 + \uparrow 1/2 + \uparrow 1/6$	 

Table J-7: Pagination Diagrams (Sheet 24 of 41)

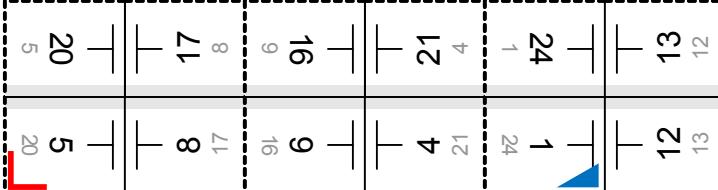
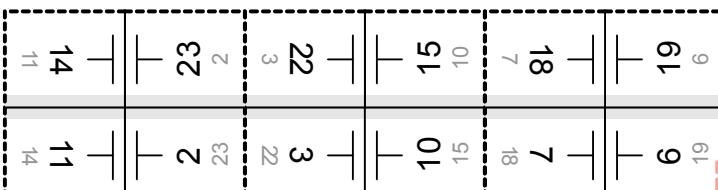
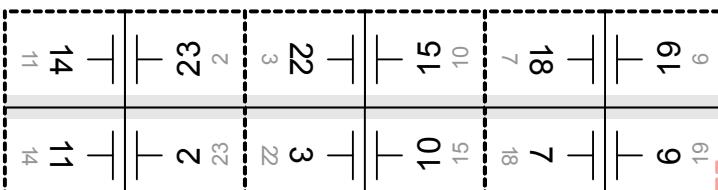
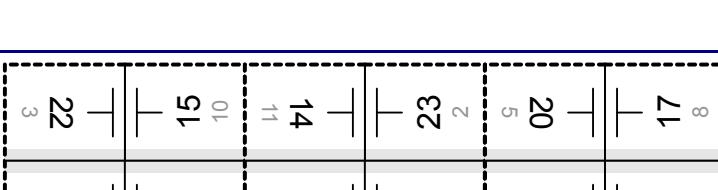
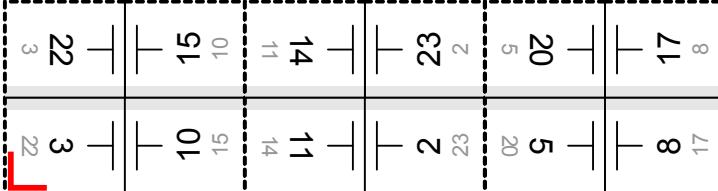
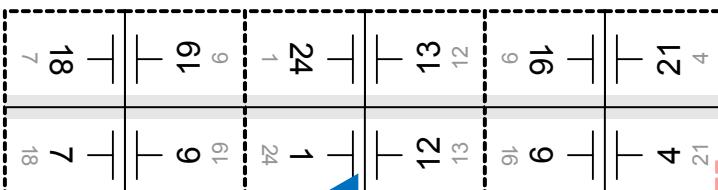
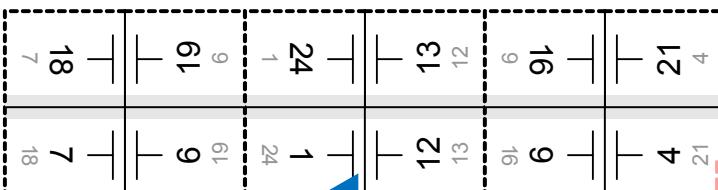
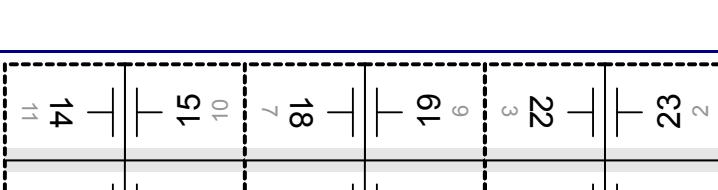
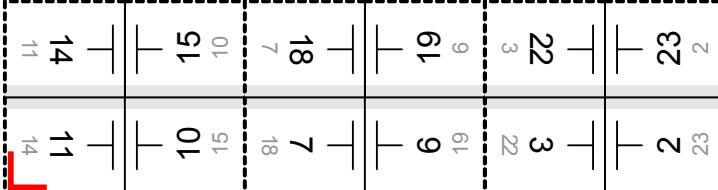
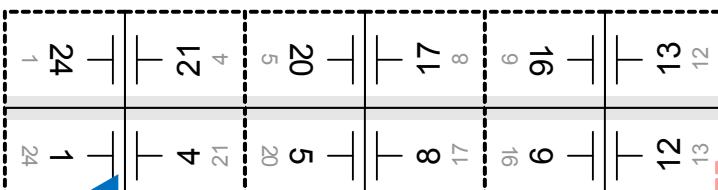
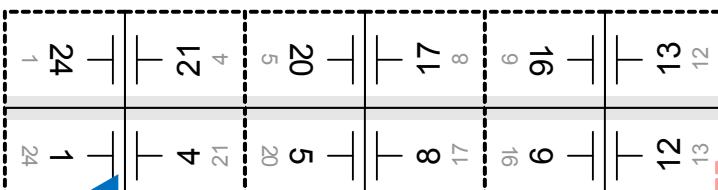
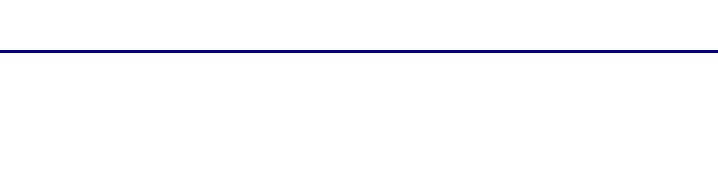
XJDF Fold Catalog	Grid Size	Front Side (Lay Side)		Back Side (Non-lay Side)	
	Folding Sequence				
F24-4	6x2				
	$\uparrow^{1/3} \downarrow^{1/3} \downarrow^{1/6} + \uparrow^{1/2}$				
F24-5	6x2				
	$\uparrow^{1/3} \uparrow^{1/3} \downarrow^{1/6} + \uparrow^{1/2}$				
F24-6	6x2				
	$\uparrow^{1/6} \downarrow^{1/6} \uparrow^{1/6}$ $\downarrow^{1/6} \uparrow^{1/6} + \uparrow^{1/2}$				

Table J-7: Pagination Diagrams (Sheet 25 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)
Folding Sequence		Back Side (Non-lay Side)
F24-7	6x2	
		$\uparrow^1/3 + \uparrow^1/2 + \uparrow^1/3$ $\downarrow^1/6$
F24-8	3x4	
		$\uparrow^1/3 \downarrow^1/3 + \uparrow^1/2 \downarrow^1/4$

Table J-7: Pagination Diagrams (Sheet 26 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	
		Folding Sequence	Back Side (Non-lay Side)
F24-9	3x4		<p>The front side of a 3x4 grid folded into a 24-page document. The grid is 4 columns wide and 6 rows high. The sequence of folds is indicated by dashed lines and numbers. A red L-shaped cutout is shown at the bottom-left corner.</p>
		$\uparrow^{2/3} \uparrow^{1/3} + \uparrow^{1/2} \downarrow^{1/4}$	<p>The back side of a 3x4 grid folded into a 24-page document. The grid is 4 columns wide and 6 rows high. The sequence of folds is indicated by dashed lines and numbers. Red squares are located at the bottom-right corner.</p>

Table J-7: Pagination Diagrams (Sheet 27 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)												Back Side (Non-lay Side)												
		Folding Sequence																								
F24-10	3x4																									

The diagram illustrates the page layout for the Front Side (Lay Side) and Back Side (Non-lay Side) of an A3 sheet folded into 24 pages (F24-10). The grid size is 3x4.

Front Side (Lay Side):

- Top Row:** Pages 11, 14, 16, 17 (top row of a 4x6 grid).
- Second Row:** Pages 14, 16, 17, 8 (middle row of a 4x6 grid).
- Third Row:** Pages 23, 2, 16, 17 (bottom row of a 4x6 grid).
- Fourth Row:** Pages 2, 23, 5, 4 (leftmost column of a 4x6 grid).
- Fifth Row:** Pages 21, 20, 5, 4 (second column of a 4x6 grid).
- Sixth Row:** Pages 21, 20, 5, 4 (third column of a 4x6 grid).
- Seventh Row:** Pages 4, 21, 20, 5 (fourth column of a 4x6 grid).

A red L-shaped arrow indicates the folding sequence: fold up-left, fold up-right, fold down-left, fold down-right.

Back Side (Non-lay Side):

- Top Row:** Pages 10, 15, 18, 7 (top row of a 4x6 grid).
- Second Row:** Pages 10, 15, 18, 7 (middle row of a 4x6 grid).
- Third Row:** Pages 22, 3, 3, 22 (bottom row of a 4x6 grid).
- Fourth Row:** Pages 22, 3, 3, 22 (leftmost column of a 4x6 grid).
- Fifth Row:** Pages 3, 22, 22, 3 (second column of a 4x6 grid).
- Sixth Row:** Pages 22, 3, 3, 22 (third column of a 4x6 grid).
- Seventh Row:** Pages 22, 3, 3, 22 (fourth column of a 4x6 grid).

A blue triangle at the bottom left indicates the folding sequence: fold up-left, fold up-right, fold down-left, fold down-right.

Table J-7: Pagination Diagrams (Sheet 28 of 41)

XJDF Fold Catalog		Grid Size	Front Side (Lay Side)													
Folding Sequence			Back Side (Non-lay Side)													
F24-11	4x3			10 15	15	14 11	11	14 11	10 15	15	14 11	11	14 11	10 15	15	
			+ \uparrow^1_4	8 17	17 8	20 5	5 20	21 4	24 1	21 4	24 1	21 4	24 1	21 4	24 1	
F28-1	7x2		+ $\uparrow^1_7 \downarrow^1_7 \uparrow^1_7 \downarrow^1_7 \uparrow^1_7 \downarrow^1_7 + \uparrow^1_2$	12 13	13 12	16 9	9 16	10 19	11 18	10 19	11 18	10 19	11 18	10 19	11 18	
				23 22	22 21	27 26	27 26	23 22	22 21	27 26	27 26	23 22	22 21	27 26	27 26	
F32-1	16x1			10 23	23 10	26 7	7 26	31 31	2 31	18 15	15 18	14 19	19 14	30 3	3 30	6 27
				12 21	21 12	28 5	5 28	29 29	4 29	20 13	13 20	16 17	17 16	32 1	1 32	8 25
				12 21	21 12	28 5	5 28	29 29	4 29	20 13	13 20	16 17	17 16	32 1	1 32	8 25

Table J-7: Pagination Diagrams (Sheet 29 of 41)

Table J-7: Pagination Diagrams (Sheet 30 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	
		Folding Sequence	Back Side (Non-lay Side)
F32-4	4x4		
		$\uparrow^1/2 + \uparrow^1/2 +$ $\uparrow^1/4 + \uparrow^1/4$	

Table J-7: Pagination Diagrams (Sheet 31 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	
		Folding Sequence	Back Side (Non-lay Side)
F32-5	4x4		
		$\uparrow^{1/2} + \uparrow^{1/2} +$ $\downarrow^{1/4} + \downarrow^{1/4}$	

Table J-7: Pagination Diagrams (Sheet 32 of 41)

Table J-7: Pagination Diagrams (Sheet 33 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)							
		Folding Sequence				Back Side (Non-lay Side)			
F32-7	4x4								

$\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4} +$
 $\uparrow^{1/2} \downarrow^{1/4}$

Table J-7: Pagination Diagrams (Sheet 34 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)	
Folding Sequence		Back Side (Non-lay Side)	
F32-8	4x4		
		$\uparrow^1/2 \downarrow^{1/4} +$ $\uparrow^1/2 \downarrow^{1/4}$	

Table J-7: Pagination Diagrams (Sheet 35 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)													
		Folding Sequence						Back Side (Non-lay Side)							
F32-9	4x4														
F36-1	9x2														

Table J-7: Pagination Diagrams (Sheet 36 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)											
		Folding Sequence						Back Side (Non-lay Side)					
F36-2	6x3												

$\uparrow^1/3 \downarrow^1/3 + \uparrow^1/3$
 $\downarrow^1/3 + \uparrow^1/6$

Table J-7: Pagination Diagrams (Sheet 37 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)											
		Folding Sequence						Back Side (Non-lay Side)					
F40-1	5x4												
		$\uparrow^{1/5} \downarrow^{1/5} \uparrow^{1/5}$ $\downarrow^{1/5} + \uparrow^{1/2} \downarrow^{1/4}$						$\uparrow^{1/5} \downarrow^{1/5} \uparrow^{1/5}$ $\downarrow^{1/5} + \uparrow^{1/2} \downarrow^{1/4}$					

Table J-7: Pagination Diagrams (Sheet 38 of 41)

XJDF Fold Catalog	Grid Size	Front Side (Lay Side)											
		Folding Sequence						Back Side (Non-lay Side)					
F48-1	6x4												

$\uparrow^1/3 \downarrow^1/3 + \uparrow^1/4$
 $\downarrow^1/6 + \uparrow^1/4 \downarrow^1/4$

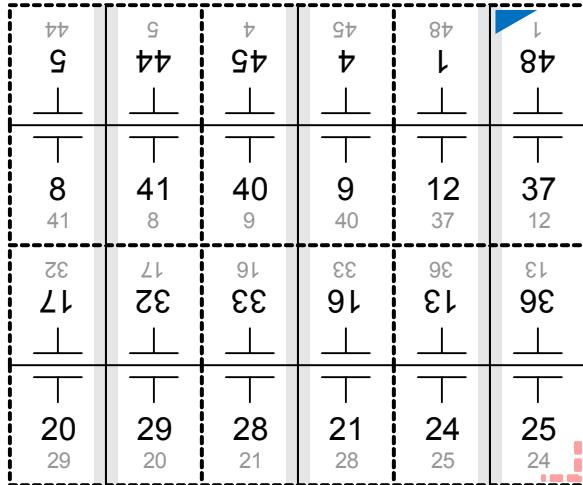
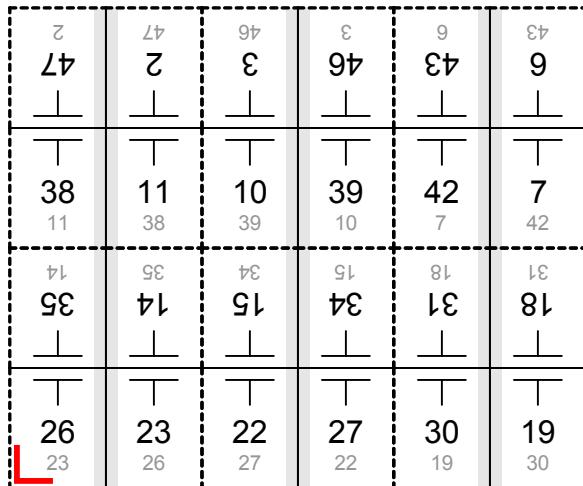


Table J-7: Pagination Diagrams (Sheet 39 of 41)

Table J-7: Pagination Diagrams (Sheet 40 of 41)

Table J-7: Pagination Diagrams (Sheet 41 of 41)

Appendix K Resolving RunList/@Directory and FileSpec/@URL URI References

This appendix describes the detailed semantics of resolving **RunList/@Directory** and any associated **FileSpec/@URL** URI references in **RunList**.

K.1 Semantics of the RunList/@Directory Attribute

The **@Directory** Attribute defines a directory where the files that are associated with this **RunList** SHOULD be copied to or from. If **@Directory** is specified, it SHALL be an Absolute URI [RFC3986] that implicitly also specifies a Base URI that is used to resolve any relative URI Attribute in the **RunList** structure. As such, **@Directory** SHALL start with a URL scheme, such as "*file*" or "*ftp*", MAY contain an authority, such as "*//any.com*" and SHOULD contain an absolute path that ends with a "/" to indicate a directory.¹ For example: "*file://any.com/pub/doc-archives/*" or "*file:///pub/doc-archives/*".

If **@Directory** is not specified, the Absolute URI that specifies the directory in which the **XJDF** file resides is used as the Base URI to resolve each relative **@URI** Attribute in the **RunList**.

XJDF After determining the Base URI depending on the presence or absence of **RunList/@Directory** as described above, each **@URL** Attribute in a **RunList** (e.g., **RunList/FileSpec/@URL** or **InsertSheet/Layout/Media/QualityControlResult/FileSpec/@URL**) is used in combination with the Base URI to form the Resolved URI as follows according to one of the following mutually exclusive patterns.²

- 1 **RunList @URL** starts with a scheme (token ending with ":" (e.g., "*file:*" or "*cid:*")):³
 - Resolved URI = the entire **RunList** URL (and the Base URI is ignored).
- 2 **RunList @URL** starts with an authority/host (starts with "//" (e.g., "*//www.cip4.org*")):
 - Resolved URI = the Base URI scheme, followed by the **RunList** URL authority/host followed by its absolute path (which MAY be empty).
- 3 **RunList @URL** starts with an absolute path (starts with "/" (e.g., "*/pub/document-archives*")):
 - Resolved URI = Base URI scheme and its authority (if any) followed by the **RunList** URL absolute path.
- 4 **RunList @URL** starts with a relative path (starts with something other than "/" (e.g., "*foo.pdf*", "./*folder/foo.pdf*", ".../*foo.pdf*", etc.)):
 - Resolved URI = Base URI scheme, its authority (if any), and its absolute path (if any) up to and including the right-most "/", followed by the **RunList @URL** relative path with ".", ".." and "./" segments removed.

The above algorithm is only a summary. See [RFC3986] for the detailed algorithm. See [FileURL] for examples.

-
1. According to [RFC3986] section 5.2 "Resolve Relative References to Absolute Form", the characters following the right-most "/" if any, are removed from the Base URI, in order to resolve a Relative URI with the Base URI. So be sure to end the **@Directory** value with a "/" to make it clear that **@Directory** is a reference to a directory and not a file, and to ensure that the last path segment won't get removed in resolving the URI reference.
 2. The Resolved URI is formed assuming that URI query and fragments are *not* used in **XJDF**.
 3. In order to improve interoperability and to simplify implementation, **XJDF** follows the strict-parsing rules of [RFC3986] so that even if the **FileSpec/@URL** Attribute starts with the same scheme as the Base URI, the entire URL values is always interpreted as an Absolute URI and always replaces the Base URI to form the Resolved URI. This strict rule is especially important for interoperability.

Appendix L References

Throughout this specification references to other documents are indicated by short symbolic names inside square brackets, (e.g., [ICC.1]). Implementers need to read and conform to such referenced documents when implementing a part of this specification with such a reference. The reader is directed to this Document References section to find the full title, date, source and availability of all such references.

Table L-1: References (Sheet 1 of 13)

Term	Definition
[Adb-TN5044]	<p><i>Adobe Technical note 5044</i></p> <p>Date: 24 May 1996 Produced by: Adobe Systems Inc. Available at: http://partners.adobe.com/public/developer/en/ps/sdk/5044.ColorSep_Conv.pdf</p>
[AdsML]	<p><i>AdsML 1.0 Specification & Schema</i></p> <p>Date: 17 May 2004 Produced by: AdsML Technical Working Group Available at: http://www.adsml.org</p>
[CCIR601-2]	<p><i>CCIR Recommendation 601-2</i> <i>Encoding Parameters of Digital Television for Studios, 1990, Volume XI — Part 1, Broadcasting Service (Television), pp. 95-104.</i></p> <p>Date: 1990 Produced by: International Telecommunication Union Available at: International Telecommunication Union, General Secretariat — Sales Section, Place des Nations, CH-1211 Geneva 20 (Switzerland)</p>
[CGATS.12/1]	<p><i>CGATS.12/1</i> <i>Graphic technology — Prepress digital data exchange — Use of PDF for composite data — Part 1: Complete exchange (PDF/X-1).</i></p> <p>Date: 14 October 1999 Produced by: Committee for Graphic Arts Technologies Standards (NPES serves as the American National Standards Institute (ANSI) secretariat to CGATS.) Available at: The publication is available in hardcopy only and may be ordered via a form at http://www.npes.org/standards/Standards-Technical-OrderForm.pdf.</p>
[CGATS.20-2002]	<p><i>CGATS.20-2002</i> <i>Graphic technology - Variable data printing exchange using PPML and PDF (PPML/VDX).</i></p> <p>Date: 2002 Produced by: Committee for Graphic Arts Technologies Standards (NPES serves as the American National Standards Institute (ANSI) secretariat to CGATS.) Available at: The publication is available in hardcopy only and may be ordered via a form at http://www.npes.org/standards/Standards-Technical-OrderForm.pdf.</p>
[CIE 15:2004]	<p><i>CIE 15:2004</i> <i>Colorimetry, 3rd Edition.</i></p> <p>Date: 2004 Produced by: Commission Internationale de l'Eclairage International (CIE) Available at: http://www.cie.co.at</p>
[ColorPS]	<p><i>Color Separation Conventions for PostScript Language Programs</i> <i>Technical Note #5044</i></p> <p>Date: 24 May 1996 Produced by: Adobe Systems Inc. Available at: http://partners.adobe.com/asn/developer/pdfs/tn/5044.ColorSep_Conv.pdf</p>

Table L-1: References (Sheet 2 of 13)

Term	Definition
[DCS2.0]	<i>Document Color Separation (DCS), version 2.0</i> Date: Revised May 1995 Produced by: Adobe Software Inc. Available at: http://www.npes.org/standards/Tools/DCS20Spec.pdf .
[distparm]	<i>Tech note 5151</i> <i>Acrobat Distiller Parameters</i> Date: August 2002 Produced by: Adobe Systems, Inc. Available at: http://partners.adobe.com/misc/search.html
[ECMA]	<i>ECMA Code of Folding Carton Styles</i> Date: - Produced by: European Carton Makers Association Available at: http://www.ecma.org/download/orderformpublications.pdf
[FEFCO]	<i>FEFCO European Federation of Corrugated Board Manufacturers</i> Date: - Produced by: European Federation of Corrugated Board Manufacturers Available at: http://www.fefco.org
[FileURL]	<i>CIP4 Application Note — Use of the File URL in JDF</i> Date: August 2003 Produced by: CIP4 Organization Available at: http://www.cip4.org
[FIRST]	<i>Flexographic Image Reproduction Specifications & Tolerances (FIRST)</i> <i>Second Edition</i> Date: November 1999 Produced by: Flexography Technical Association Available at: http://www.fta-ffta.org .
[GRACoL]	<i>General Requirements for Applications in Commercial offset Lithography (GRACoL)</i> <i>Version 6.0</i> Date: June 2002 Produced by: IDEAlliance (formerly Graphic Communications Association) Available at: http://www.gracol.com .
[iana-mt]	<i>IANA Registry of MIME Media Types</i> Available at: http://www.iana.org/assignments/media-types
[iana-os]	<i>IANA Registry of Operating System Names</i> Available at: http://www.iana.org/assignments/operating-system-names
[ICC.1]	<i>Specification ICC.1:2004-10</i> <i>File Format for Color Profiles, Version 4.2.0.0</i> Date: 2004 Produced by: International Color Consortium (ICC) Available at: http://www.color.org/icc_specs2.xalter
[IEEE754]	<i>IEEE 754-1985</i> <i>Standard for Binary Floating-Point Arithmetic</i> Date: 1985 Produced by: IEEE Available at: http://grouper.ieee.org/groups/754/

Table L-1: References (Sheet 3 of 13)

Term	Definition
[IEEE1284]	<p><i>IEEE 1284-2000</i> <i>IEEE Standard Signaling Method for a Bidirectional Parallel Peripheral Interface for Personal Computers</i></p> <p>Date: 2000 Produced by: IEEE Available at: http://standards.ieee.org/catalog/olis/busarch.html</p>
[IEEE-ISTO 5100.1-2001]	<p><i>IEEE-ISTO 5100.1-2001</i> <i>IPP/1.1: finishings attribute values extension</i></p> <p>Date: February 5, 2001 Produced by: IEEE-ISTO Available at: ftp://ftp.pwg.org/pub/pwg/standards/pwg5100.1.pdf, .doc, .rtf</p>
[IEEE-ISTO 5100.2-2001]	<p><i>IEEE-ISTO 5100.2-2001</i> <i>IPP/1.0 & 1.1: "Output-bin" attribute extensions</i></p> <p>Date: February 7, 2001 Produced by: IEEE-ISTO Available at: ftp://ftp.pwg.org/pub/pwg/standards/pwg5100.2.pdf, .doc, .rtf</p>
[IEEE-ISTO 5100.3-2001]	<p><i>IEEE-ISTO 5100.3-2001</i> <i>Production Printing Attributes - Set1</i></p> <p>Date: February 17, 2001 Produced by: IEEE-ISTO Available at: ftp://ftp.pwg.org/pub/pwg/standards/pwg5100.3.pdf, .doc, .rtf</p>
[IEEE-ISTO 5100.4-2001]	<p><i>IEEE-ISTO 5100.4-2001</i> <i>Override Attributes for Documents and Pages</i></p> <p>Date: February 7, 2001 Produced by: IEEE-ISTO Available at: ftp://ftp.pwg.org/pub/pwg/standards/pwg5100.4.pdf, .doc, .rtf</p>
[ifra]	<p><i>IfraTrack Specification</i> <i>Ifra Special Report 6.21.2, Version 2.0</i></p> <p>Date: June 1998 Produced by: Ifra Available at: http://www.ifra.com/</p>
[ISO5-3:1995]	<p><i>ISO 5-3:1995</i> <i>Photography -- Density measurements -- Part 3: Spectral conditions.</i></p> <p>Date: 1995 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISOstore/store.html</p>
[ISO5-4:1995]	<p><i>ISO 5-4:1995</i> <i>Photography -- Density measurements -- Part 4: Geometric conditions for reflection density.</i></p> <p>Date: 1995 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISOstore/store.html</p>
[ISO216:1975]	<p><i>ISO 216:1975</i> <i>Writing paper and certain classes of printed matter -- Trimmed sizes -- A and B series.</i></p> <p>Date: 1975 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISOstore/store.html</p>

Table L-1: References (Sheet 4 of 13)

Term	Definition
[ISO269:1985]	<p><i>ISO 269:1985</i> <i>Correspondence envelopes -- Designation and sizes</i> Date: 1985 Produced by: ISO Available at:http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO2470:1999]	<p><i>ISO 2470:1999</i> <i>Paper, board and pulps -- Measurement of diffuse blue reflectance factor (ISO brightness).</i> Date: 1999 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO2471:1998]	<p><i>ISO 2471:1998</i> <i>Paper and board—Determination of opacity (paper backing)—Diffuse reflectance method.</i> Date: 1998 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO2846-1:1997]	<p><i>ISO 2846-1:1997</i> <i>Graphic technology - Colour and transparency of ink sets for four-colour-printing - Part 1: Sheet-fed and heat-set web offset lithographic printing.</i> Date: 1997 Produced by: ISO Available at:http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO3166-1:1997]	<p><i>ISO 3166-1:1997</i> <i>Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes.</i> Date: 1997 Produced by: ISO Available at:http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO4217:2001]	<p><i>ISO 4217:2001</i> <i>Codes for the representation of currencies and funds</i> Date: 2001 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO8254-1:1999]	<p><i>ISO 8254-1:1999</i> <i>Paper and board -- Measurement of specular gloss -- Part 1: 75 degree gloss with a converging beam, TAPPI method</i> Date: 1999 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO8601:2004]	<p><i>ISO 8601:2004</i> <i>Data elements and interchange formats - Information interchange - Representation of dates and times.</i> Date: 2004 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>

Table L-1: References (Sheet 5 of 13)

Term	Definition
[ISO12639:2004]	<p><i>ISO 12639:2004</i> <i>Graphic technology - Prepress digital data exchange — Tag image file format for image technology (TIFF/IT)</i></p> <p>Date: 2004 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO12647-2:2004]	<p><i>ISO 12647-2:2004</i> <i>Graphic technology - Process control for the production of half-tone colour separations, proof and production prints - Part 2: Offset lithographic processes.</i></p> <p>Date: 2004 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO12647-2:2013]	<p><i>ISO 12647-2:2013</i> <i>Graphic technology - Process control for the production of half-tone colour separations, proof and production prints - Part 2: Offset lithographic processes.</i></p> <p>Date: 2013 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO13655:1996]	<p><i>ISO 13655:1996</i> <i>Graphic technology -- Spectral measurement and colorimetric computation for graphic arts images.</i></p> <p>Date: 1996 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO14977:1996]	<p><i>ISO 14977:1996(E)</i> <i>Information technology -- Syntactic metalanguage -- Extended BNF</i></p> <p>Date: 1996 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO15076-1:2005]	<p><i>ISO 15076-1:2005</i> <i>Image technology colour management -- Architecture, profile format and data structure -- Part 1: Based on ICC.1:2004-10. See [ICC.1].</i></p> <p>Date: 2005 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO15930-1:2001]	<p><i>ISO 15930-1:2001</i> <i>Graphic technology — Prepress digital data exchange — Use of PDF — Part 1: Complete exchange using CMYK data (PDF/X-1 and PDF/X-1a).</i></p> <p>Date: 2001 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>
[ISO15930-4:2003]	<p><i>ISO 15930-4:2003</i> <i>Graphic technology — Prepress digital data exchange — Use of PDF — Part 1: Complete exchange using CMYK data (PDF/X-1 and PDF/X-1a).</i></p> <p>Date: 2003 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISostore/store.html</p>

Table L-1: References (Sheet 6 of 13)

Term	Definition
[ISO15930-5:2003]	<p><i>ISO 15930-2:2003</i></p> <p><i>Graphic technology — Prepress digital data exchange — Use of PDF — Part 2: Partial exchange of printing data (PDF/X-2).</i></p> <p>Date: 2003 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISOstore/store.html</p>
[ISO15930-3:2002]	<p><i>ISO 15930-3:2002</i></p> <p><i>Graphic technology — Prepress digital data exchange — Use of PDF — Part 3: Complete exchange suitable for colour-managed workflows (PDF/X-3).</i></p> <p>Date: 2002 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISOstore/store.html</p>
[ISO15930-6:2003]	<p><i>ISO 15930-6:2003</i></p> <p><i>Graphic technology — Prepress digital data exchange — Use of PDF — Part 3: Complete exchange suitable for colour-managed workflows (PDF/X-3).</i></p> <p>Date: 2003 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISOstore/store.html</p>
[ISO15930-7:2010]	<p><i>ISO 15930-7:2010</i></p> <p><i>Graphic technology — Prepress digital data exchange — Use of PDF — Complete exchange suitable for colour-managed workflows (PDF/X-4).</i></p> <p>Date: 2010 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISOstore/store.html</p>
[ISO15930-8:2010]	<p><i>ISO 15930-8:2010</i></p> <p><i>Graphic technology — Prepress digital data exchange — Use of PDF — Complete exchange suitable for colour-managed workflows (PDF/X-5).</i></p> <p>Date: 2010 Produced by: ISO Available at: http://www.iso.ch/iso/en/prods-services/ISOstore/store.html</p>
[JIS P 0138:1998]	<p><i>JIS P 0138:1998</i></p> <p>Writing paper and certain classes of printed matter -- Trimmed sizes -- A and B series.</p> <p>Date: 1998 Produced by: JIS Available at: http://www.webstore.jsa.or.jp/webstore or google title</p>
[japancolor]	<p><i>Japan Color 2001</i></p> <p>Date: 2001 Produced by: Japan Printing Machinery Manufacturers Association, Office of JNC for TC130 Available at: Call (81) 03-3434-4661</p>
[JDF15]	<p><i>Job Definition Format 1.5</i></p> <p>Date: 2013 Produced by: International Cooperation for Integration of Processes in Prepress, Press and Postpress (CIP4) Available at: http://www.cip4.org</p>

Table L-1: References (Sheet 7 of 13)

Term	Definition
[K&R]	<i>C Programming Language</i> , by Brian W. Kernighan and Dennis M. Ritchie <i>Second Edition</i> Date: March 22, 1988 Produced by: Prentice Hall Available at: (Book only. Look for ISBN 0131103628.)
[macbinary]	<i>Macintosh Binary Transfer Format ("MacBinary III") Standard Proposal.</i> Date: December 1996 Produced by: Macintosh Internet Developer Association Available at: http://www.lazerware.com/formats/
[opentypefont]	<i>OpenType specification</i> <i>v.1.4</i> Date: 11 October 2002 Produced by: Microsoft Corporation Available at: http://www.microsoft.com/typography/specs/
[PDF1.6]	<i>PDF reference : Adobe portable document format version 1.6 / Adobe Systems Incorporated.</i> <i>Version 1.6</i> Date: November 2004 Produced by: Addison-Wesley Available at: http://partners.adobe.com/public/developer/pdf/index_reference.html
[PJTF]	<i>The Portable Job Ticket Format</i> <i>Version 1.1</i> Date: 2 April 1999 Produced by: Adobe Systems Inc. Available at: http://partners.adobe.com/asn/developer/pdfs/tn/5620.pjtf.pdf
[PPD]	<i>Adobe PostScript Printer Description File Format Specification</i> <i>Version 4.3</i> Date: 9 February 1996 Produced by: Adobe Systems Inc. Available at: http://www.cip4.org/documents/technical_info/cip3v3_0.pdf
[PPF]	<i>Print Production Format</i> <i>Version 3.0</i> Date: 2 June 1998 Produced by: The International Cooperation for Integration of Prepress, Press and Postpress Available at: http://www.cip4.org/documents/technical_info/cip3v3_0.pdf
[PPML]	<i>PPML</i> <i>Personal Print Markup Language (PPML)</i> <i>Version 2.1</i> Produced by: Print On Demand Initiative (PODi) Available at: http://www.podi.org
[PrintTalk]	<i>PrintTalk Implementation</i> <i>Version 1.1</i> Produced by: PrintTalk Consortium Available at: http://www.printtalk.org/

Table L-1: References (Sheet 8 of 13)

Term	Definition
[PS]	<p><i>PostScript Language Reference (Redbook) Third Edition</i></p> <p>Date: — Produced by: Adobe Systems, Inc. Available at: http://partners.adobe.com/asn/developer/pdfs/tm/PLRM.pdf</p>
[PWG]	<p><i>The Printer Working Group</i></p> <p>Date: — Produced by: IEEE-ISTO Available at: http://www.pwg.org</p>
[PWGFINMIB]	<p><i>Printer Finishing MIB</i> (draft-ietf-printmib-finishing-16.txt — work in progress.)</p> <p>Date: February 2003 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: IETF Internet-Drafts have a six month life-time. They are available at: https://datatracker.ietf.org/public/pidtracker.cgi</p>
[Quark]	See http://www.quark.com .
[RFC1738]	<p><i>RFC 1738</i> <i>Uniform Resource Locators (URL)</i></p> <p>Date: 1994 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC1741]	<p><i>RFC 1741</i> <i>MIME Content Type for BinHex Encoded Files, by Faltstrom, P., Crocker, D. and Fair, E.</i></p> <p>Date: December 1994 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC1759]	<p><i>RFC 1759</i> <i>Printer MIB, Version 2.0 by Smith, R., Wright, F., Hastings, T., Zilles, S. and Gyllenskog, J.</i></p> <p>Date: June 2003 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC1766]	<p><i>RFC 1766</i> <i>Tags for the Identification of Languages, by H. Alvestrand.</i></p> <p>Date: March 1995 Produced by: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC1950]	<p><i>RFC 1950</i> <i>ZLIB Compressed Data Format Specification version 3.3, by P. Deutsch.</i></p> <p>Date: May 1996 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>

Table L-1: References (Sheet 9 of 13)

Term	Definition
[RFC1951]	<p><i>RFC 1951</i> <i>DEFLATE Compressed Data Format Specification version 1.3, by Deutsch, P.</i></p> <p>Date: May 1996 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC1952]	<p><i>RFC 1952</i> <i>GZIP file format specification version 4.3, by Deutsch, P.</i></p> <p>Date: May 1996 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC1977]	<p><i>RFC 1977</i> <i>PPP BSD Compression Protocol, by Schryver, V.</i></p> <p>Date: August 1996 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC2045]	<p><i>RFC 2045</i> <i>Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, by Freed, N. and Borenstein, N. (Updated by RFC2184, RFC2231)</i></p> <p>Date: November 1996 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC2046]	<p><i>RFC 2046</i> <i>Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types, by Freed, N. and Borenstein, N. (Updated by RFC2646)</i></p> <p>Date: November 1996 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC2183]	<p><i>RFC 2183</i> <i>Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field</i></p> <p>Date: August 1997 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC2231]	<p><i>RFC 2231</i> <i>MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations</i></p> <p>Date: November 1997 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>

Table L-1: References (Sheet 10 of 13)

Term	Definition
[RFC2368]	<p><i>RFC 2368</i> <i>The mailto URL scheme</i> by P. Hoffman, L. Masinter and J. Zawinski.</p> <p>Date: July 1998 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC2387]	<p><i>RFC 2387</i> <i>The MIME Multipart/Related Content-type</i>, by Levinson, E.</p> <p>Date: August 1998 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC2392]	<p><i>RFC 2392</i> <i>Content-ID and Message-ID Uniform Resource Locators</i>, by Levinson, E.</p> <p>Date: August 1998 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC2616]	<p><i>RFC 2616</i> <i>Hypertext Transfer Protocol — HTTP/1.1</i></p> <p>Date: June 1999 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC2822]	<p><i>RFC 2822</i> <i>Internet Message Format</i></p> <p>Date: April 2001 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC2911]	<p><i>RFC 2911</i> <i>Internet Printing Protocol/1.1: Model and Semantics</i>, by T. Hastings, R. Herriot, R. deBry, S. Isaacson and P. Powell.</p> <p>Date: September 2000 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC3302]	<p><i>RFC 3302</i> <i>Tag Image File Format (TIFF) — image/tiff MIME Sub-type Registration</i>, by Parsons, G., Rafferty, J.</p> <p>Date: September 2002 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>

Table L-1: References (Sheet 11 of 13)

Term	Definition
[RFC3381]	<p><i>RFC 3381</i></p> <p><i>Internet Printing Protocol (IPP): Job Progress Attributes</i> by T. Hastings, H. Lewis and R. Bergman.</p> <p>Date: September 2002 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC3382]	<p><i>RFC 3382</i></p> <p><i>Internet Printing Protocol (IPP): The 'collection' attribute syntax</i> by R. deBry, R. Herriot, T. Hastings, K. Ocke and P. Zehler.</p> <p>Date: September 2002 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC3548]	<p><i>RFC 3548</i></p> <p><i>The Base16, Base32, and Base64 Data Encodings</i>, by S. Josefsson</p> <p>Date: July 2003 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC3966]	<p><i>RFC 3966</i></p> <p><i>The tel URI for Telephone Numbers</i> by H. Schulzrinne</p> <p>Date: December 2004 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC3986]	<p><i>FC 3986</i></p> <p>Uniform Resource Identifier (URI): Generic Syntax by T. Berners-Lee, R. Fielding and L. Masinter</p> <p>Date: January 2005 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[RFC3987]	<p><i>RFC 3987</i></p> <p><i>Internationalized Resource Identifiers (IRIs)</i>, by M. Duerst and M. Suignard</p> <p>Date: January 2005 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: All IETF (Internet Engineering Task Force) RFCs (Request for Comments) are available at RFC Database search: http://www.rfc-editor.org/rfcsearch.html.</p>
[SNAP]	<p><i>Specifications for Newsprint Advertising Production (SNAP)</i></p> <p>Date: 2000 Produced by: Printing Industries of America, Inc. (SNAP Committee) Available at: http://www.gain.net/store/item.cfm?productid=488</p>
[SSL3]	<p><i>SSL Specification</i></p> <p>Netscape, The SSL Protocol, Version 3 (Text version 3.0.2), November 1996. http://wp.netscape.com/eng/ssl3/draft302.txt</p>

Table L-1: References (Sheet 12 of 13)

Term	Definition
[TAPPI T480]	TAPPI T480 <i>Specular Gloss of Paper and Paperboard at 75 Degrees, Test Method T 519 om-99</i> Date: not stated Produced by: TAPPI. Available at: http://www.tappi.org
[TAPPI T519]	TAPPI T519 <i>Diffuse Opacity of Paper (d/0 paper backing), Test Method T 519 om-02</i> Date: not stated Produced by: TAPPI. Available at: http://www.tappi.org
[TAPPI T527]	TAPPI T527 <i>Color of Paper and Paperboard (d/0, C/2), Test Method T 527 om-02</i> Date: not stated Produced by: TAPPI. Available at: http://www.tappi.org
[TAPPI T560]	TAPPI T560 <i>CIE Whiteness and Tint of Paper and Paperboard (Using d/0°, Diffuse Illumination and Normal Viewing), Test Method T 560 wd-03</i> Date: not stated Produced by: TAPPI. Available at: http://www.tappi.org
[TIFF6]	<i>TIFF Revision 6.0</i> Date: June 1992 Produced by: Adobe Systems, Inc. Available at: http://partners.adobe.com/asn/tech/tiff/specification.jsp
[TIFFPS]	<i>Adobe Photoshop TIFF Technical Notes</i> Date: March 2002 Produced by: Adobe Systems, Inc. Available at: http://partners.adobe.com/asn/tech/tiff/specification.jsp
[truetypefont]	<i>TrueType font file and TrueType Open specification</i> Date: August 1995 Produced by: Microsoft Corporation Available at: http://www.microsoft.com/typography/specs/
[type1font]	<i>Adobe Type 1 Font Format</i> <i>Adobe Systems, Inc.</i> Date: 1990 Produced by: Addison-Wesley Publishing Company, Inc. Available at: http://partners.adobe.com/asn/developer/pdfs/tn/T1_SPEC.PDF
[Unicode5.0]	<i>The Unicode Standard, Version 5.0,</i> Date: November 3, 2006 Produced by: The Unicode Consortium Available at: http://www.unicode.org/book/aboutbook.html

Table L-1: References (Sheet 13 of 13)

Term	Definition
[uuencode]	<p><i>Unix Uuencode, The Single UNIX® Specification, Version 2</i> (Converts binary into the local character set that is suitable to pass through email systems.)</p> <p>Date: 1997 Produced by: The Open Group Available at: http://www.opengroup.org/onlinepubs/007908799/xcu/uuencode.html</p>
[UPNP]	<p><i>Printer Device and Printer Basic Service Version 1.0</i></p> <p>Date: 2002 Produced by: Universal Plug N Play Forum Available at: http://www.upnp.org/standardizeddcps/printer.asp</p>
[URI]	<p><i>URIs, URLs, and URNs: Clarifications and Recommendations 1.0) Version (W3C Recommendation of 21 September 2001)</i></p> <p>Date: 21 September 2001 Produced by: World Wide Web Consortium (W3C) Available at: http://www.w3.org/TR/2001/NOTE-uri-clarification-20010921/</p>
[WINZip]	<p><i>APPNOTE.TXT — .ZIP File Format Specification Version 5.2</i></p> <p>Date: 16 July 2003 Produced by: PKWARE Inc. Available at: http://www.pkware.com/products/enterprise/white_papers/appnote.html</p>
[XML]	<p><i>Extensible Markup Language (XML) 1.0 (Fifth Edition) Version (W3C Recommendation of 8 December 2009)</i></p> <p>Date: 8 December 2009 Produced by: World Wide Web Consortium (W3C) Available at: http://www.w3.org/TR/2008/REC-xml-20081126/</p>
[XMLNS]	<p><i>Namespaces in XML 1.0 (Third Edition) Version (W3C Recommendation of 8 December 2009)</i></p> <p>Date: 8 December 2009 Produced by: World Wide Web Consortium (W3C) Available at: http://www.w3.org/TR/2009/REC-xml-names-20091208/</p>
[XMLSchema]	<p><i>XML Schema Part 0+1+2: Primer, Structures and Datatypes Version (W3C Recommendation of 28 Oct 2004)</i></p> <p>Date: 28 October 2004 Produced by: World Wide Web Consortium (W3C) XML Schema working group Available at: http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/, http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/ and http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/.</p>
[XPath]	<p><i>XML Path Language (XPath) 2.0 (Second Edition) Version W3C Recommendation 14 December 2010</i></p> <p>Date: 14 December 2010 Produced by: World Wide Web Consortium (W3C) Available at: http://www.w3.org/TR/2010/REC-xpath20-20101214/</p>
[X.509]	<p><i>Public-Key Infrastructure (X.509) (pkix) Version 1.1</i></p> <p>Date: 14 February 2008 Produced by: IETF. Available at: http://www.ietf.org/html.charters/pkix-charter.html.</p>

Appendix L References

Appendix M XJDF/CIP4 Hole Pattern Catalog

The following table defines the specifics of the predefined holes in **HoleMakingParams** and **HoleMakingParams**.

Notes:

- 1 All patterns are centered on the Sheet along the process edge.
- 2 Process Edge is always defined relative to a portrait orientation of the medium, regardless of the orientation of the printed image or processing path.
- 3 Thumbcuts are available in various standard shapes (labeled “No. *N*” where *N* is minimally ranging from 2..7). “No. 3” seems to be the most widely used.
- 4 Single thumbcuts appear always in the center of the process edge.
- 5 Oval shape holes actually look sometimes more like rectangular holes with rounded corners.

Sources:

- 1 [PWGFINMIB]

Naming Scheme:

Table M-1: Naming Scheme for Hole Patterns

Name	Description
General	<m i>: m = metric (millimeter is used), i = imperial (inch, where 1 inch = 25.4 mm)
Ring Binding	R<#holes><m i>-<variant> Example: R2m-DIN = RingBind, 2 hole, metric, DIN
Plastic Comb	P<pitch><m i>-<shape>-<#thumbcuts>t Example: P16:9m-round-0t = Plastic Comb, 9/16" pitch (16:9), round, no thumbcut
Wire Comb	W<pitch><m i>-<shape>-<#thumbcuts>t Example: W2:1i-square-1t = Wire Comb, 1/2" pitch (2:1), square, one thumbcut
Coil/Spiral	C<pitch><m i>-<shape>-<#thumbcuts>t Example: C9.5m-round-0t = Coil, 9.5 mm, round, no thumbcut
Special	S<#holes> Example: S1-generic

Table M-2: Hole Details for R2 Series (Sheet 1 of 2)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt (!)	Default Process Edge	Usage Notes	Source Standard
RING BINDING (R...)										
2 Holes (R2...)										
R2-generic	Generic request of a 2-hole pattern	2	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	34.02 (\equiv 12 mm)	Left	See note (7).	N/A
R2m-DIN	DIN 2-hole MIB: 6 = two-HoledIN and 10 = twoHoleMetric	2	●	5.5 ± 0.1 mm	80 ± 0.1 mm	7 or 11 ± 0.3 mm 7 mm for blocks of <= 15 mm thick	31.18 (\equiv 11 mm)	Left	A4 and A5	DIN 5005:1991 DIN 821:1973
R2m-ISO	ISO 2-hole MIB: 6 = two-HoledIN and 10 = twoHoleMetric	2	●	6 ± 0.5 mm	80 ± 0.5 mm	12 ± 1 mm	34.02 (\equiv 12 mm)	Left	Also used in Japan	ISO 838:1974 (E)
R2m-MIB	Printer Finishing MIB twoHoleDIN and twoHoleMetric	2	●	5-8 mm	80 ± 0.5 mm	4.5 - 13 mm	31.18 (\equiv 11 mm)	Left	Printer Finishing MIB	
R2i-US-a	US 2-hole, Variant A MIB: 4 = twoHoleDUSTop and 12 = twoHoleUS-Side	2	●	0.2 - 0.32"	2.75"	0.18 - 0.51"	29.25 (\equiv 13/32")	Left for letter Top for ledger	Printer Finishing MIB	

Table M-2: Hole Details for R2 Series (Sheet 2 of 2)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Pattern Axis Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt{!}	Default Process Edge	Usage Notes	Source Standard
R2i-US-b	US 2-hole, Variant B	2	●	0.2-0.5" default: 5/16"	6"	0.25" + ½ diameter range: 6/16" - 1/2"	29.25 (≈ 13/32")	Left		

Table M-3: Hole Details for R3 and R4 Series (Sheet 1 of 2)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Pattern Axis Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt{!}	Default Process Edge	Usage Notes	Source Standard
RING BINDING (R...)										
3 Holes (R3...)	Generic request of a 3-hole pattern.	3	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	29.25 (≈ 13/32")	Left	See note (7).	N/A
R3i-US	US 3-hole MIB: 5 = threeHoleUS	3	●	std: 5/16" mg: 0.2-0.5" typ: 1/4", 9/32", 11/32", 3/8", 13/32", 1/2"	4.25"	0.25" + ½ diameter range: 6/16" - 1/2"	29.25 (≈ 13/32")	Left		Printer Finishing MIB
4 Holes (R4...)										
R4-generic	Generic request of a 4-hole pattern.	4	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	31.18 (≈ 11 mm)	Left	See note (7).	N/A

Table M-3: Hole Details for R3 and R4 Series (Sheet 2 of 2)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Pattern Axis Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt (!)	Default Process Edge	Usage Notes	Source Standard
R4m-DIN-A4	DIN 4-hole for A4	4	●	5.5 ± 0.1 mm	80 ± 0.1 mm	7 or 11 ± 0.3 mm 7 mm for blocks of 15 mm or less	31.18 (≤ 11 mm)	Left	A4	DIN 5005:1991
R4m-DIN-A5	DIN 4-hole for A5	4	●	5.5 ± 0.1 mm	45-65-45 mm	7 or 11 ± 0.3 mm 7 mm for blocks of 15 mm or less	31.18 (≤ 11 mm)	Left	A5	DIN 821:1973
R4m-swedish	Swedish 4-hole MIB: 11 = swed- ish4Hole	4	●	5 - 8 mm	21-70-21 mm	4.5 - 13 mm	31.18 (≤ 11 mm)	Left for A4 Top for A3	A4, A3	Printer Finishing MIB
R4i-US	US 4-hole	4	●	0.2 - 0.5" std: 5/16" typ: 1/4", 9/ 32", 11/32", 3/8", 13/32", 1/2"	1.375-4.25- 1.375"	0.25" + ½ diameter range: 6/16" - 1/ 2"	29.25 (≤ 0.25" + ½ x 5/ 16" = 13/32")	Left		

Table M-4: Hole Details for R5 and R6 Series (Sheet 1 of 3)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Pattern Axis Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt (!)	Default Process Edge	Usage Notes	Source Standard
RING BINDING (R...)										
5 Holes (R5...)	Generic request of a 5-hole pattern.	5	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	29.25 (≤ 13/32")	Left	See note (7).	N/A

Table M-4: Hole Details for R5 and R6 Series (Sheet 2 of 3)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Pattern Axis Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt()	Default Process Edge	Usage Notes	Source Standard
R5i-US-a	US 5-hole, Variant A MIB: 13 = fiveHoleUS	5	●	0.2 - 0.32"	2-2.25-2.25-2"	0.18 - 0.51"	29.25 ($\equiv 13/32"$)	Left for letter Top for ledger	Printer Finishing MIB	
R5i-US-b	US 5-hole, Variant B	5	●	0.2 - 0.5" std: 5/16" typ: 1/4", 9/ 32", 11/32", 3/8", 13/32", 1/2"	0.75-3.5- 3.5-0.75" 0.375 - 0.5"	0.25" + ½ diameter 0.375 - 0.5"	29.25 ($\equiv 0.25" + \frac{1}{2} \times 5/16" = 13/32"$)	Left		
R5i-US-c	Combination of R2i-US-a and R3i-US	5	●	0.2 - 0.5" std: 5/16" typ: 1/4", 9/ 32", 11/32", 3/8", 13/32", 1/2"	1.25-3-3- 1.25"	0.25" + ½ diameter 0.375 - 0.5"	29.25 ($\equiv 0.25" + \frac{1}{2} \times 5/16" = 13/32"$)	Left		
6 Holes (R6...)										
R6-generic	Generic request of a 6-hole pattern.	6	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	31.18 ($\equiv 11$ mm)	Left for A4/A5 Top for A3	See note (7).	N/A
R6m-4h2s	Norwegian 4-hole (round) mixed with 2 slots (rectangular) MIB: 16 = norweg6Hole	6	H: S: S:	● ■ ■	Holes: 5 - 8 mm Slots: 10 x 5.5 mm	4 holes/2 slots Pattern: H-H-S-S-H-H 64-18.5-75-18.5-64 mm	4.5 - 13 mm 31.18 ($\equiv 11$ mm)	Left for A4 Top for A3	Printer Finishing MIB	

Table M-4: Hole Details for R5 and R6 Series (Sheet 3 of 3)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Pattern Axis Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt (!)	Default Process Edge	Usage Notes	Source Standard
R6m-DIN-A5	DIN 6-hole for A5	6	●	5.5 ± 0.1 mm	37.5-7.5-65-7.5-37.5 mm	7 or 11 ± 0.3 mm 7 mm for blocks of <= 15 mm thick	31.18 (≈ 11 mm)	Left	Only used with A5	DIN 5005:1991

Table M-5: Hole Details for R7 and R11 Series (Sheet 1 of 2)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Pattern Axis Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt (!)	Default Process Edge	Usage Notes	Source Standard
RING BINDING (R....)										
7 Holes (R7...)										
R7-generic	Generic request of a 7-hole pattern.	7	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	29.25 (≈ 13/32")	Left for letter Top for ledger	See note (7).	N/A
R7i-US-a	US 7-hole, Variant A MIB: 14 = seven-HoleUS	7	●	0.2 - 0.32"	1-1-2.25- 2.25-1-1"	0.18 - 0.51"	29.25 (≈ 13/32")	Left for letter Top for ledger	Printer Finishing MIB	
R7i-US-b	US 7-hole, Bell/AT&T Systems. Combination of R3i-US, R4i-US, R5i-US-b	7	●	0.2 - 0.5" std: 5/16" typ: 1/4", 9/ 32", 11/32", 3/8", 13/32", 1/2"	0.75-1.375- 2.125- 2.125- 0.375 - 0.5"	0.25" + ½ diameter 0.375 - 0.5"	29.25 (≈ 0.25" + ½ x 5/ 16" = 13/32")	Left for letter Top for ledger		

Table M-5: Hole Details for R7 and R11 Series (Sheet 2 of 2)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Pattern Axis Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt (!)	Default Process Edge	Usage Notes	Source Standard
R71-US-c	US 7-hole, Variant C	7	●	0.2 - 0.5" std: 5/16" typ: 1/4", 9/ 32", 11/32", 3/8", 13/32", 1/2"	1.25-0.875- 2.125- 0.875-1.25"	0.25" + 1/2 dia- ter 0.375 - 0.5"	29.25 (\equiv 13/32")	Left for letter Top for ledger		
R11m-7h4s	7-hole (round) mixed with 4 slots (rectangular) MIB: 15 = mixed7H4S	11	● ■	H: S: mm	Holes: 5 - 8 mm Slots: 12 x 6 mm	7 holes/ 2slots Pattern: H- S-H-H-S- H-S-H-H- S-H	4.5 - 13 mm 31.18 (\equiv 11 mm)	Left for A4 Top for A3	Printer Finishing MIB	

Table M-6: Hole Details for P, W, C and S Series (Sheet 1 of 3)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent:	Pattern Geometry	Pattern Axis Offset from Process Edge	Pattern Axis Offset from Process Edge in pt (!)	XJDF Default Pattern Axis Offset from Process Edge	Default Process Edge	Usage Notes	Source Standard
PLASTIC COMB BINDING (P...)											
P16_9i-rect-0t	US spacing, no thumbcut MIB: 9 = nineteen-HoleUS	A4: 21 Letter: 19	■	5/16" x 1/8" (8 x 3.2 mm)	9/16"	3/16"	13.54 ($\cong 0.188"$)	Left		Printer Finishing MIB	

Table M-6: Hole Details for P, W, C and S Series (Sheet 2 of 3)

Table M-6: Hole Details for P, W, C and S Series (Sheet 3 of 3)

XJDF Hole Pattern Catalog ID	Description	# Holes	Hole Shape	Hole Extent	Pattern Geometry	Pattern Axis Offset from Process Edge	XJDF Default Pattern Axis Offset from Process Edge in pt ()	Default Process Edge	Usage Notes	Source Standard
S-generic	Generic request of a hole pattern with an arbitrary or unknown number of holes (e.g., an inline shotgun).	1 or more	●	5 - 13 mm 0.2-0.51"	N/A	N/A	N/A	Any	N/A	N/A
S1-generic	Generic request of a hole pattern with 1 hole.	1	●	5 - 13 mm 0.2-0.51"	N/A	N/A	N/A	Any	N/A	N/A

Appendix N FileSpec Attributes and Container Subelement

The purpose of this appendix is to give a series of use cases with examples for the use of the Attributes of **FileSpec**: *@MimeType*, *@URL*, *@Compression* and the **FileSpec**/*@ContainerRef* Subelement. These use cases include container packaging files, such as tar, zip and Multipart/Related files and container compression and encoding files, each of which require one or more *@ContainerRef* Subelements to link one **FileSpec** with its container **FileSpec**.

N.1 Examples of Attribute Values of FileSpec

Table N-1 shows a number of use cases and the corresponding values for the *@MimeType*, *@URL* and *@Compression* Attributes. Each *@ContainerRef* Element points to the **FileSpec** shown on the next row in the table. The use cases are arranged in order of increasing complexity.

Note: All of the *@URL* examples in this appendix for **FileSpec** Resources that are not contained in other files are Absolute URIs, so that the complication of resolving **FileSpec**/*@URI* with **RunList**/*@Directory* is not considered. Of course, the *@URL* examples for **FileSpec** Resources that are contained in other files SHALL all be Relative URIs (relative to the Base URI that is defined to be the Absolute URI of where the XJDF Consumer extracted the container file) as the XJDF spec requires (see the *@URL* description at Section 8.61, “FileSpec” on page 662).

Table N-1: Use Cases showing MimeType, URL and Compression Attribute Values (Sheet 1 of 2)

Description of Use Case	Mime Type	URL	Compression
1.) Single a.pdf PDF file, no compression	application/pdf	ftp://www.any.com/share/a.pdf	
2.) Single a.pdf PDF file, with Gzip compression	application/pdf	a.pdf	Gzip
Container FileSpec	Gzip	ftp://www.any.com/a.gz	
3.) Single a.pdf PDF file, no compression, but Base64 encoded	application/pdf	a.pdf	Base64
Container FileSpec	Base64	ftp://www.any.com/a.mme	
4.) Single PDF file, no compression, but BinHex encoded into a BinHex file	application/pdf	a.pdf	BinHex
Container FileSpec	application/mac-binhex40	ftp://www.any.com/a.hqx	
5.) Single a.pdf PDF file with ZLIB compression in b.zip ZIP file (containing one or more files)	application/pdf	a.pdf	ZLIB
Container FileSpec	application/zip	ftp://www.any.com/b.zip	
6.) Single a.pdf PDF file compressed by Deflate in a b.zip with one or more files, and the b.zip packaging file itself is Base64 encoded as b.mme. To read, unencode, then uncompress. To write, compress, then encode.	application/pdf	a.pdf	Deflate
Container FileSpec	application/zip	b.zip	Base64
Container FileSpec	Base64	ftp://www.any.com/b.mme	

Table N-1: Use Cases showing MimeType, URL and Compression Attribute Values (Sheet 2 of 2)

Description of Use Case	Mime Type	URL	Compression
7.) Single myFiles/myPicture.jpg file in myNestedZip.zip file with one or many files, but the myNestedZip.zip is itself zipped into c.zip.	image/jpeg	myFiles/myPicture.jpg	Deflate
Container FileSpec	application/zip	myNestedZip.zip	Deflate
Container FileSpec	application/zip	ftp://www.any.com/c.zip	
8.) Single a.pdf PDF file which is ZLIB compressed, in a c.zip with one or many files which is contained in a tar file and compressed with ZLIB into a.tar.gz file.	application/pdf	a.pdf	ZLIB
Container FileSpec	application/zip	c.zip	
Container FileSpec	Tar ^a	d.tar	ZLIB
Container FileSpec	GZip	ftp://www.any.com/d.tar.gz	

- a. The UNIX Tar file packaging format is not registered with IANA as a MIME media type, so CIP4 has assigned the “Tar” file type to it for use in the **FileSpec/@MimeType** Attribute.

N.2 Corresponding XML examples

The above use case examples are represented in XML as follows:

Example N-1: FileSpec #1

Single a.pdf PDF file, No Compression:

TBD 2.x Example.

```
<FileSpec Class="Parameter" ID="F1" Status="Available"
          MimeType="application/pdf" URL="ftp://www.any.com/share/a.pdf"/>
```

Example N-2: FileSpec #2

Single a.pdf PDF file, with Gzip compression:

TBD 2.x Example.

```
<FileSpec Class="Parameter" ID="F1" Status="Available"
          Compression="Gzip" MimeType="application/pdf" URL="a.pdf">
    <Container>
        <FileSpec MimeType="Gzip" URL="ftp://www.any.com/a.gz"/>
    </Container>
</FileSpec>
```

Example N-3: FileSpec #3

Single a.pdf PDF file, no compression, but Base64 encoded:

TBD 2.x Example.

```
<FileSpec Class="Parameter" ID="F1" Status="Available"
          Compression="Base64" MimeType="application/pdf" URL="a.pdf">
    <Container>
        <FileSpec MimeType="Base64" URL="ftp://www.any.com/a.mme"/>
    </Container>
</FileSpec>
```

Example N-4: FileSpec #4

Single PDF file, no compression, but BinHex encoded:

TBD 2.x Example.

```
<FileSpec Class="Parameter" ID="F1" Status="Available"
    Compression="BinHex"MimeType="application/pdf" URL="a.pdf">
<Container>
    <FileSpecMimeType="application/mac-binhex40"
        URL="ftp://www.any.com/a.hqx"/>
</Container>
</FileSpec>
```

Example N-5: FileSpec #5

Single a.pdf PDF file, in b.zip ZIP file containing one or more files:

TBD 2.x Example.

```
<FileSpec Class="Parameter" ID="F1" Status="Available"
    Compression="ZLIB"MimeType="application/pdf" URL="a.pdf">
<Container>
    <FileSpecMimeType="application/zip" URL="ftp://www.any.com/b.zip"/>
</Container>
</FileSpec>
```

Example N-6: FileSpec #6

Single a.pdf PDF file, in a b.zip with one or more files, and the b.zip packaging file itself is Base64 encoded as b.mme. To read, decode, then decompress. To write, compress, then encode.

TBD 2.x Example.

```
<FileSpec Class="Parameter" ID="F1" Status="Available"
    Compression="Deflate"MimeType="application/pdf" URL="a.pdf">
<Container>
    <FileSpec Compression="Base64"MimeType="application/zip" URL="b.zip">
        <Container>
            <FileSpecMimeType="Base64" URL="ftp://www.any.com/b.mme"/>
        </Container>
    </FileSpec>
</Container>
</FileSpec>
```

Example N-7: FileSpec #7

Single myFiles/myPicture.jpg file in myNestedZip.zip file with one or many files, but the myNestedZip.zip is itself zipped into c.zip

TBD 2.x Example.

```
<FileSpec Class="Parameter" ID="F1" Status="Available"
    Compression="Deflate"MimeType="image/jpeg" URL="myFiles/myPicture.jpg">
<Container>
    <FileSpec Compression="Deflate"MimeType="application/zip"
        URL="myNestedZip.zip">
        <Container>
            <FileSpecMimeType="application/zip"
                URL="ftp://www.any.com/c.zip"/>
        </Container>
    </FileSpec>
</Container>
</FileSpec>
```

```
</Container>
</FileSpec>
```

Example N-8: FileSpec #8

Single a.pdf PDF file, which is ZLIB compressed in a c.zip with one or many files which is contained in a tar file and compressed with ZLIB into a.tar.gz file.:.

TBD 2.x Example.

```
<FileSpec Class="Parameter" ID="F1" Status="Available"
    Compression="ZLIB"MimeType="application/pdf" URL="a.pdf">
<Container>
    <FileSpec MimeType="application/zip" URL="c.zip">
        <Container>
            <FileSpec Compression="ZLIB" MimeType="Tar" URL="d.tar">
                <Container>
                    <FileSpec MimeType="GZip"
                        URL="ftp://www.any.com/d.tar.gz"/>
                </Container>
            </FileSpec>
        </Container>
    </FileSpec>
</Container>
</FileSpec>
</Container>
</FileSpec>
```

N.3 Additional examples showing Partitioning of FileSpec

This section has additional examples of container files and various schemes of Partitioning.

Example N-9: FileSpec #9

Package b.zip contains multiple pdf files a.pdf, b.pdf etc.

TBD 2.x Example.

```
<FileSpec Class="Parameter" Status="Available" ID="ID_002"
    MimeType="application/zip"
    URL="ftp://www.any.com/b.zip"/>
<FileSpec Class="Parameter" Status="Available" ID="A_FILE"
    Compression="Deflate" MimeType="application/pdf"
    URL="a.pdf">
    <Container>
        <FileSpecRef rRef="ID_002"/>
    </Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="B_FILE"
    Compression="Deflate" MimeType="application/pdf"
    URL="b.pdf">
    <Container>
        <FileSpecRef rRef="ID_002"/>
    </Container>
</FileSpec>
```

Example N-10: FileSpec #10

Package b.zip contains two pdf files a.pdf, b.pdf and a tiff, c.tiff used by a Partitioned Resource

TBD 2.x Example.

```
<FileSpec Class="Parameter" Status="Available" ID="ID_003"
    MimeType="application/zip" URL="ftp://www.any.com/b.zip"/>
```

```

<FileSpec Class="Parameter" Status="Available" ID="ALL_FILES"
    Compression="Deflate"MimeType="application/pdf"
    PartIDKeys="PartVersion">
<Container>
    <FileSpecRef rRef="ID_003"/>
</Container>
<FileSpec PartVersion="English" URL="a.pdf"/>
<FileSpec PartVersion="French" URL="b.pdf"/>
<FileSpec MimeType="application/tif" PartVersion="German" URL="c.tif"/>
</FileSpec>

```

Example N-11: FileSpec #11

Single a.pdf PDF file, in b.zip which is contained in c.tar file:

TBD 2.x Example.

```

<FileSpec Class="Parameter" Status="Available" ID="ID_004_TAR" MimeType="Tar"
    URL="ftp://www.any.com/c.tar"/>
<FileSpec Class="Parameter" Status="Available" ID="ID_004_ZIP"
    MimeType="application/zip" URL="b.zip">
<Container>
    <FileSpecRef rRef="ID_004_TAR"/>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="C_FILE"
    Compression="Deflate" MimeType="application/pdf" URL="a.pdf">
<Container>
    <FileSpecRef rRef="ID_004_ZIP"/>
</Container>
</FileSpec>

```

Example N-12: FileSpec #11.1 — No Partitioning

Multiple files in several zip's contained in a tar file, various examples with and without Partitioning,

So the file layout looks like:

```

e.tar
  c.zip
    a.pdf
    b.pdf
  d.zip
    a.pdf
    b.pdf

```

TBD 2.x Example.

```

<FileSpec Class="Parameter" Status="Available" ID="ID_005_TAR"
    MimeType="Tar" URL="ftp://www.any.com/e.tar"/>
<FileSpec Class="Parameter" Status="Available" ID="ID_005_ZIP_C"
    MimeType="application/zip" URL="c.zip">
<Container>
    <FileSpecRef rRef="ID_005_TAR"/>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="ID_005_ZIP_D"
    MimeType="application/zip" URL="d.zip">
<Container>
    <FileSpecRef rRef="ID_005_TAR"/>
</Container>
</FileSpec>

```

```

<FileSpec Class="Parameter" Status="Available" ID="A_ENGLISH_FILE"
    Compression="Deflate"MimeType="application/pdf"
    URL="a.pdf">
<Container>
    <FileSpecRef rRef="ID_005_ZIP_C"/>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="B_ENGLISH_FILE"
    Compression="Deflate"MimeType="application/pdf"
    URL="b.pdf">
<Container>
    <FileSpecRef rRef="ID_005_ZIP_C"/>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="A_GERMAN_FILE"
    Compression="Deflate"MimeType="application/pdf"
    URL="a.pdf">
<Container>
    <FileSpecRef rRef="ID_005_ZIP_D"/>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="B_GERMAN_FILE"
    Compression="Deflate"MimeType="application/pdf"
    URL="b.pdf">
<Container>
    <FileSpecRef rRef="ID_005_ZIP_D"/>
</Container>
</FileSpec>

```

Example N-13: FileSpec #11.2 — Intermediate container Partitioned

TBD 2.x Example.

```

<FileSpec Class="Parameter" Status="Available" ID="ID_005_TAR" MimeType="Tar"
    URL="ftp://www.any.com/e.tar"/>
<FileSpec Class="Parameter" Status="Available" ID="ID_005_ZIPS"
    MimeType="application/zip" PartIDKeys="PartVersion">
    <Container>
        <FileSpecRef rRef="ID_005_TAR"/>
    </Container>
    <FileSpec PartVersion="English" URL="c.zip"/>
    <FileSpec PartVersion="German" URL="d.zip"/>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="A_ENGLISH_FILE"
    Compression="Deflate"MimeType="application/pdf"
    URL="a.pdf">
    <Container>
        <FileSpecRef rRef="ID_005_ZIPS">
            <Part PartVersion="English"/>
        </FileSpecRef>
    </Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="B_ENGLISH_FILE"
    Compression="Deflate"MimeType="application/pdf"
    URL="b.pdf">
    <Container>
        <FileSpecRef rRef="ID_005_ZIPS">
            <Part PartVersion="English"/>
        </FileSpecRef>
    </Container>
</FileSpec>

```

```

</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="A_GERMAN_FILE"
    Compression="Deflate" MimeType="application/pdf"
    URL="a.pdf">
<Container>
    <FileSpecRef rRef="ID_005_ZIPS">
        <Part PartVersion="German"/>
    </FileSpecRef>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="B_GERMAN_FILE"
    Compression="Deflate" MimeType="application/pdf"
    URL="b.pdf">
<Container>
    <FileSpecRef rRef="ID_005_ZIPS">
        <Part PartVersion="German"/>
    </FileSpecRef>
</Container>
</FileSpec>

```

Example N-14: FileSpec #11.3 — the pdf is Partitioned

TBD 2.x Example.

```

<FileSpec Class="Parameter" Status="Available" ID="ID_005_TAR"
    MimeType="Tar" URL="ftp://www.any.com/e.tar"/>
<FileSpec Class="Parameter" Status="Available" ID="ID_005_ZIP_C"
    MimeType="application/zip" URL="c.zip">
<Container>
    <FileSpecRef rRef="ID_005_TAR"/>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="ID_005_ZIP_D"
    MimeType="application/zip" URL="d.zip">
<Container>
    <FileSpecRef rRef="ID_005_TAR"/>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="ALL_FILES"
    Compression="Deflate" PartIDKeys="PartVersion DocIndex">
    <!-- English Files -->
    <FileSpec PartVersion="English">
        <Container>
            <FileSpecRef rRef="ID_005_ZIP_C"/>
        </Container>
        <!-- English File A -->
        <FileSpec DocIndex="1" MimeType="application/pdf" URL="a.pdf"/>
        <!-- English File B -->
        <FileSpec DocIndex="2" MimeType="application/pdf" URL="b.pdf"/>
    </FileSpec>
    <!-- German Files -->
    <FileSpec PartVersion="German">
        <Container>
            <FileSpecRef rRef="ID_005_ZIP_D"/>
        </Container>
        <!-- German File A -->
        <FileSpec DocIndex="1" MimeType="application/pdf" URL="a.pdf"/>
        <!-- German File B -->
        <FileSpec DocIndex="2" MimeType="application/pdf" URL="b.pdf"/>
    </FileSpec>

```

```
</FileSpec>
</FileSpec>
```

Example N-15: FileSpec #11.3a — the pdf is Partitioned, Different File Layout

As above but the file layout is not reflected in the container structure, the files are intermingled

TBD 2.x Example.

```
<FileSpec Class="Parameter" Status="Available" ID="ID_005_TAR"
          MimeType="Tar" URL="ftp://www.any.com/e.tar"/>
<FileSpec Class="Parameter" Status="Available" ID="ID_005_ZIP_C"
          MimeType="application/zip" URL="c.zip">
<Container>
    <FileSpecRef rRef="ID_005_TAR"/>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="ID_005_ZIP_D"
          MimeType="application/zip" URL="d.zip">
<Container>
    <FileSpecRef rRef="ID_005_TAR"/>
</Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="ALL_FILES"
          Compression="Deflate" MimeType="application/pdf"
          PartIDKeys="PartVersion DocIndex">
    <!-- English Files -->
    <FileSpec PartVersion="English">
        <!-- English File A -->
        <FileSpec DocIndex="1" URL="a.pdf">
            <Container>
                <FileSpecRef rRef="ID_005_ZIP_C"/>
            </Container>
        </FileSpec>
        <!-- English File B -->
        <FileSpec DocIndex="2" URL="a.pdf">
            <Container>
                <FileSpecRef rRef="ID_005_ZIP_D"/>
            </Container>
        </FileSpec>
    </FileSpec>
    <!-- German Files -->
    <FileSpec PartVersion="German">
        <!-- German File A -->
        <FileSpec DocIndex="1" URL="b.pdf">
            <Container>
                <FileSpecRef rRef="ID_005_ZIP_C"/>
            </Container>
        </FileSpec>
        <!-- German File B -->
        <FileSpec DocIndex="2" URL="b.pdf">
            <Container>
                <FileSpecRef rRef="ID_005_ZIP_D"/>
            </Container>
        </FileSpec>
    </FileSpec>
</FileSpec>
```

Example N-16: FileSpec #11.4 — Both Partitioned

TBD 2.x Example.

```

<FileSpec Class="Parameter" Status="Available" ID="ID_005_TAR" MimeType="Tar"
          URL="ftp://www.any.com/e.tar"/>
<FileSpec ID="ID_005_ZIPS" MimeType="application/zip"
          PartIDKeys="PartVersion">
    <Container>
        <FileSpecRef rRef="ID_005_TAR"/>
    </Container>
    <FileSpec PartVersion="English" URL="c.zip"/>
    <FileSpec PartVersion="German" URL="d.zip"/>
</FileSpec>
<FileSpec Compression="Deflate" ID="ALL_FILES"
          PartIDKeys="PartVersion DocIndex">
    <!-- English Files -->
    <FileSpec PartVersion="English">
        <Container>
            <FileSpecRef rRef="ID_005_ZIPS">
                <Part PartVersion="English"/>
            </FileSpecRef>
        </Container>
        <!-- English File A -->
        <FileSpec DocIndex="1" MimeType="application/pdf" URL="a.pdf"/>
        <!-- English File B -->
        <FileSpec DocIndex="2" MimeType="application/pdf" URL="b.pdf"/>
    </FileSpec>
    <!-- German Files -->
    <FileSpec PartVersion="German">
        <Container>
            <FileSpecRef rRef="ID_005_ZIPS">
                <Part PartVersion="German"/>
            </FileSpecRef>
        </Container>
        <!-- German File A -->
        <FileSpec DocIndex="1" MimeType="application/pdf" URL="a.pdf"/>
        <!-- German File B -->
        <FileSpec DocIndex="2" MimeType="application/pdf" URL="b.pdf"/>
    </FileSpec>
</FileSpec>

```

Example N-17: FileSpec #12

Multiple PDF and TIFF files in several zip's contained in a tar file. Use all PDF files in c.zip, using the **FileSpec/@FileFormat** mechanism and just Pictures/TIFS/a.pdf in d.zip. File layout looks like:

```

e.tar
  c.zip
    a.pdf
    a.tif
    b.pdf
    b.tif
  d.zip
    PDFS/a.pdf
    PDFS/b.pdf
    Pictures/TIFS/a.pdf
    Pictures/TIFS/b.pdf

```

TBD 2.x Example.

```
<FileSpec Class="Parameter" Status="Available" ID="ID_005_TAR" MimeType="Tar"
          URL="ftp://www.any.com/e.tar"/>
<FileSpec Class="Parameter" Status="Available" ID="ID_005_ZIP_C"
          MimeType="application/zip" URL="c.zip">
  <Container>
    <FileSpecRef rRef="ID_005_TAR"/>
  </Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="ID_005_ZIP_D"
          MimeType="application/zip" URL="d.zip">
  <Container>
    <FileSpecRef rRef="ID_005_TAR"/>
  </Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="PDF_FILES"
          Compression="Deflate" FileFormat="%s.pdf" FileTemplate="all"
          MimeType="application/pdf">
  <Container>
    <FileSpecRef rRef="ID_005_ZIP_C"/>
  </Container>
</FileSpec>
<FileSpec Class="Parameter" Status="Available" ID="Pictures"
          Compression="Deflate" URL="Pictures/TIFS/a.pdf">
  <Container>
    <FileSpecRef rRef="ID_005_ZIP_D"/>
  </Container>
</FileSpec>
```

N.4 Example of an Intent Job Ticket with a doubly nested ZIP packaging file

Here is a complete example of an intent Job ticket using **DeliveryParams** with a doubly nested packaging file. The example shows a myPictures.jpg file that is contained in myNestedZip.zip file which is contained in myZip.zip file:

Example N-18: Intent Job Ticket

TBD 2.x Example.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="FileSpecProposal01" JobID="bookJob"
      JobPartID="bookJob-1" Status="Waiting" Type="Product" Version="1.4">
  <ResourcePool>
    <ArtDeliveryIntent Class="Intent" ID="FileSpecProposal02" Status="Draft">
      <ArtDelivery ArtDeliveryType="DigitalMedia">
        <RunListRef rRef="FileSpecProposal05"/>
      </ArtDelivery>
    </ArtDeliveryIntent>
    <RunList ID="FileSpecProposal05" Class="Parameter" Status="Available">
      <LayoutElement>
        <FileSpec Compression="Deflate" MimeType="image/jpeg"
                  URL="myFiles/myPicture.jpg">
          <Container>
            <FileSpecRef rRef="ID_002"/>
          </Container>
        </FileSpec>
      </LayoutElement>
    </RunList>
    <Component Amount="100" Class="Quantity" ComponentType="FinalProduct"
               DescriptiveName="FileSpec Test" ID="FileSpecProposal03">
  </JDF>
```

```

        Status="Unavailable"/>
<FileSpec Class="Parameter" Status="Available" ID="ID_001"
         MimeType="application/zip" URL="http://www.CIP4.org/myZip.zip"/>
<FileSpec Class="Parameter" Status="Available" Compression="Deflate"
          ID="ID_002" MimeType="application/zip" URL="myNestedZip.zip">
<Container>
  <FileSpecRef rRef="ID_001"/>
</Container>
</FileSpec>
</ResourcePool>
<ResourceLinkPool>
  <ComponentLink Amount="100" Usage="Output" rRef="FileSpecProposal03"/>
  <ArtDeliveryIntentLink Usage="Input" rRef="FileSpecProposal02"/>
</ResourceLinkPool>
</JDF>

```

N.5 AppOS and OSVersion Attributes

This section lists examples values for the following Attributes of the **FileSpec** Resource: **@AppOS** and **@OSVersion**. The listing is intended to be exhaustive for the most likely operating systems that are routinely used in **XJDF** applications. However, other operating systems and combinations MAY be used as well. When operating systems have new versions, they can be used and SHOULD follow the patterns established in this the following table.

Table N-2: AppOS and OSVersion Examples

AppOS	OSVersion	Description
Linux	2.2	Linux operation system
Mac	10.2.4	Macintosh operation system
Solaris	4.0	Sun Solaris operation system
UNIX	BSD	Berkeley UNIX
UNIX	V	System V UNIX
UNIX	V.1	System V UNIX
UNIX	V.2	System V UNIX
UNIX	V.3	System V UNIX
UNIX	PC	UNIX for the PC
Windows	95	Windows 95
Windows	98	Windows 98
Windows	NT	Windows NT
Windows	NT-5	Windows 2000
Windows	NT-5.1	XP [not yet registered by Microsoft with IANA]

Appendix O Examples

Note that these examples were generated using prototype tools and are intended to be used for general overview only. The emphasis is **not** on the individual bytes (e.g., capitalization or exact keywords). Normative examples will be provided at <http://www.CIP4.org> when available.

Note: pay special attention to XJDF tags that are **orange** and/or XJDF attribute names that are **magenta**.

O.1 Brief Example

O.1.1 Before and After Processing

Example O-1: Before Processing

This is a simple example of an XJDF that describes color conversion for one file.

TBD 2.x Example.

Example O-2: After Processing

This is a simple example of an XJDF that describes color conversion for one file after the color conversion process has been executed.

TBD 2.x Example.

O.2 Product XJDF

Example O-3: Product XJDF

The following example describe a pair of college textbooks, one teachers edition and one students edition as Product Intent. Most Intent Resources are intentionally left empty.

TBD 2.x Example.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" ID="bookTest" JobID="bookJob"
      Status="Waiting" JobPartID="ID455" Type="Product" Version="1.4">
  <ResourcePool>
    <Component Amount="100" Class="Quantity" DescriptiveName="Teacher's Book"
               ID="Link0003" Status="Unavailable" ComponentType="Sheet"/>
    <Component Amount="2000" Class="Quantity" DescriptiveName="Cover" ID="Link0005"
               Status="Unavailable" ComponentType="Sheet">
      <!--This cover is reused by both-->
    </Component>
    <LayoutIntent Class="Intent" ID="Link0006" Status="Available">
      <Dimensions DataType="XYPairSpan" Preferred="612 792"
                    Range="576 756 ~ 648 828"/>
    </LayoutIntent>
    <LayoutIntent Class="Intent" ID="Link0008" Status="Available">
      <Dimensions DataType="XYPairSpan" Preferred="612 792"
                    Range="576 756 ~ 648 828"/>
      <Pages DataType="IntegerSpan" Preferred="240"/>
    </LayoutIntent>
    <Component Amount="1000" Class="Quantity" DescriptiveName="Student's Book"
               ID="Link0011" Status="Unavailable" ComponentType="Sheet">
      <!--Students Book Intent-->
    </Component>
    <LayoutIntent Class="Intent" ID="Link0014" Status="Available">
      <Dimensions DataType="XYPairSpan" Preferred="612 792"
                    Range="576 756 ~ 648 828"/>
    </LayoutIntent>
  </ResourcePool>
</JDF>
```

```

        <Pages DataType="IntegerSpan" Preferred="198"/>
    </LayoutIntent>
</ResourcePool>
<AuditPool>
    <Created AgentName="Rainer's JDFWriter 0.2000"
             TimeStamp="2000-11-01T12:46:56+01:00"/>
</AuditPool>
<ResourceLinkPool>
    <ComponentLink Amount="1000" Usage="Output" rRef="Link0011"/>
</ResourceLinkPool>
<JDF DescriptiveName="Teacher's Edition" ID="Link0002" JobPartID="0"
      Status="Waiting" Type="Product">
    <ResourcePool>
        <Component Amount="100" Class="Quantity" DescriptiveName="Insert"
                   ID="Link0009" Status="Unavailable" ComponentType="Sheet"/>
    </ResourcePool>
    <ResourceLinkPool>
        <ComponentLink Amount="100" Usage="Output" rRef="Link0003"/>
        <ComponentLink Amount="100" Usage="Input" rRef="Link0009"/>
        <ComponentLink Amount="100" Usage="Input" rRef="Link0005"/>
    </ResourceLinkPool>
    <JDF DescriptiveName="Teacher's Insert" ID="Link0007" JobPartID="2"
          Status="Waiting" Type="Product">
        <ResourceLinkPool>
            <LayoutIntentLink Usage="Input" rRef="Link0008"/>
            <ComponentLink Amount="100" Usage="Output" rRef="Link0009"/>
        </ResourceLinkPool>
    </JDF>
</JDF>
<JDF DescriptiveName="Cover" ID="Link0004" JobPartID="1" Status="Waiting"
      Type="Product">
    <ResourceLinkPool>
        <ComponentLink Amount="2000" Usage="Output" rRef="Link0005"/>
        <LayoutIntentLink Usage="Input" rRef="Link0006"/>
    </ResourceLinkPool>
</JDF>
<JDF DescriptiveName="Student's Edition" ID="Link0010" JobPartID="3"
      Status="Waiting" Type="Product">
    <ResourcePool>
        <Component Amount="1000" Class="Quantity" DescriptiveName="Insert"
                   ID="Link0013" Status="Unavailable" ComponentType="Sheet"/>
    </ResourcePool>
    <ResourceLinkPool>
        <ComponentLink Amount="1000" Usage="Output" rRef="Link0011"/>
        <ComponentLink Amount="1000" Usage="Input" rRef="Link0013"/>
        <ComponentLink Amount="1000" Usage="Input" rRef="Link0005"/>
    </ResourceLinkPool>
    <JDF DescriptiveName="Student's Insert" ID="Link0012" JobPartID="4"
          Status="Waiting" Type="Product">
        <ResourceLinkPool>
            <ComponentLink Amount="1000" Usage="Output" rRef="Link0013"/>
            <LayoutIntentLink Usage="Input" rRef="Link0014"/>
        </ResourceLinkPool>
    </JDF>
</JDF>
</JDF>

```

O.3 Spawning and Merging

The following set of examples show an **XJDF** Job in the relevant stages of spawning and merging. One example defines a simple brochure with a cover and an insert. The Node in green emphasis, which defines the cover, is spawned, modified, and subsequently merged. Elements in red emphasis represent metadata that apply to spawning and merging.

Example O-4: 2-Component XJDF before Spawning

The following **XJDF** file describes a two-component brochure. The Resources are not fleshed out.

Note: pay special attention to **XJDF** tags that are orange and/or **XJDF** attribute names that are magenta.

TBD 2.x Example.

```
<JDF ID="SpawnTest" Type="Product" xmlns="http://www.CIP4.org/JDFSchema_1_1"
      Status="Waiting" Version="1.4" JobPartID="Part1">
  <AuditPool>
    <Created AgentName="CIP4 JDFWriter 1.0.1 beta"
             TimeStamp="2002-04-05T15:27:58+02:00"/>
  </AuditPool>
  <ResourcePool>
    <Component ID="r0043" Class="Quantity" Amount="10000"
               Status="Unavailable" ComponentType="Sheet" />
    <BindingIntent ID="r0044" Class="Intent" Status="Available">
      <BindingType Range="SaddleStitch" DataType="EnumerationSpan" />
    </BindingIntent>
    <ProductionIntent ID="r0045" Class="Intent" Status="Available">
      <PrintProcess Range="Gravure" DataType="EnumerationSpan" />
    </ProductionIntent>
    <Component ID="r0047" Class="Quantity" Status="Unavailable"
               ComponentType="Sheet" />
    <Component ID="r0051" Class="Quantity" Status="Unavailable"
               ComponentType="Sheet" />
  </ResourcePool>
  <ResourceLinkPool>
    <ComponentLink rRef="r0043" Usage="Output"/>
    <BindingIntentLink rRef="r0044" Usage="Input"/>
    <ProductionIntentLink rRef="r0045" Usage="Input"/>
    <ComponentLink rRef="r0047" Usage="Input"/>
    <ComponentLink rRef="r0051" Usage="Input"/>
  </ResourceLinkPool>
  <JDF ID="n0046" Type="Product" Status="Waiting" JobPartID="Part2"
        DescriptiveName="Cover">
    <ResourceLinkPool>
      <ComponentLink rRef="r0047" Usage="Output"/>
      <LayoutIntentLink rRef="r0048" Usage="Input"/>
      <ColorIntentLink rRef="r0049" Usage="Input"/>
    </ResourceLinkPool>
    <ResourcePool>
      <LayoutIntent ID="r0048" Class="Intent" Status="Available" />
      <ColorIntent ID="r0049" Class="Intent" Status="Available" />
    </ResourcePool>
  </JDF>
  <JDF ID="n0050" Type="Product" Status="Waiting" JobPartID="Part3"
        DescriptiveName="Insert">
    <ResourceLinkPool>
      <ComponentLink rRef="r0051" Usage="Output"/>
      <LayoutIntentLink rRef="r0052" Usage="Input"/>
      <ColorIntentLink rRef="r0053" Usage="Input"/>
    </ResourceLinkPool>
  </JDF>
</JDF>
```

```

</ResourceLinkPool>
<ResourcePool>
    <LayoutIntent ID="r0052" Class="Intent" Status="Available"/>
    <ColorIntent ID="r0053" Class="Intent" Status="Available"/>
</ResourcePool>
</JDF>
</JDF>

```

TBD 2.x Example.

```

<JDF ID="SpawnTest" Type="Product" xmlns="http://www.CIP4.org/JDFSchema_1_1"
      Status="Waiting" Version="1.4" JobPartID="Part1">
    <AuditPool>
        <Created AgentName="CIP4 JDFWriter 1.0.1 beta"
                 TimeStamp="2002-04-05T15:27:58+02:00"/>
        <Spawned URL="File:///spawn.jdf" jRef="n0046"
                 TimeStamp="2002-04-05T15:34:43+02:00"
                  NewSpawnID="Sp0057" rRefsRWCopied="r0047"/>
    </AuditPool>
    <ResourcePool>
        <Component ID="r0043" Class="Quantity" Amount="10000"
                   Status="Unavailable" ComponentType="Sheet" />
        <BindingIntent ID="r0044" Class="Intent" Status="Available">
            <BindingType Range="SaddleStitch" DataType="EnumerationSpan"/>
        </BindingIntent>
        <ProductionIntent ID="r0045" Class="Intent" Status="Available">
            <PrintProcess Range="Gravure" DataType="NameSpan"/>
        </ProductionIntent>
        <Component ID="r0047" Class="Quantity" Status="Unavailable"
                   ComponentType="Sheet" SpawnIDs="Sp0057" SpawnStatus="SpawnedRW" />
        <Component ID="r0051" Class="Quantity" Status="Unavailable"
                   ComponentType="Sheet" />
    </ResourcePool>
    <ResourceLinkPool>
        <ComponentLink rRef="r0043" Usage="Output"/>
        <BindingIntentLink rRef="r0044" Usage="Input"/>
        <ProductionIntentLink rRef="r0045" Usage="Input"/>
        <ComponentLink rRef="r0047" Usage="Input"/>
        <ComponentLink rRef="r0051" Usage="Input"/>
    </ResourceLinkPool>
    <JDF ID="n0046" Type="Product" Status="Spawned" JobPartID="Part2"
          DescriptiveName="Cover">
        <ResourceLinkPool>
            <ComponentLink rRef="r0047" Usage="Output"/>
            <LayoutIntentLink rRef="r0048" Usage="Input"/>
            <ColorIntentLink rRef="r0049" Usage="Input"/>
        </ResourceLinkPool>
        <ResourcePool>
            <LayoutIntent ID="r0048" Class="Intent" Status="Available"
                          SpawnIDs="Sp0057" SpawnStatus="SpawnedRO"/>
            <ColorIntent ID="r0049" Class="Intent" Status="Available"
                          SpawnIDs="Sp0057" SpawnStatus="SpawnedRO"/>
        </ResourcePool>
    </JDF>
    <JDF ID="n0050" Type="Product" Status="Waiting" JobPartID="Part3"
          DescriptiveName="Insert">
        <ResourceLinkPool>
            <ComponentLink rRef="r0051" Usage="Output"/>
            <LayoutIntentLink rRef="r0052" Usage="Input"/>

```

```

        <ColorIntentLink rRef="r0053" Usage="Input"/>
    </ResourceLinkPool>
    <ResourcePool>
        <LayoutIntent ID="r0052" Class="Intent" Status="Available"/>
        <ColorIntent ID="r0053" Class="Intent" Status="Available"/>
    </ResourcePool>
</JDF>
</JDF>

```

TBD 2.x Example.

```

<JDF ID="n0046" Type="Product" xmlns="http://www.CIP4.org/JDFSchema_1_1"
    Status="Waiting" SpawnID="Sp0057" Version="1.4" JobPartID="Part2"
    DescriptiveName="Cover">
    <AuditPool>
        <Created AgentName="CIP4 JDFWriter 1.0.1 beta"
            TimeStamp="2002-04-05T15:34:43+02:00"/>
    </AuditPool>
    <ResourceLinkPool>
        <ComponentLink rRef="r0047" Usage="Output"/>
        <LayoutIntentLink rRef="r0048" Usage="Input"/>
        <ColorIntentLink rRef="r0049" Usage="Input"/>
    </ResourceLinkPool>
    <ResourcePool>
        <LayoutIntent ID="r0048" Class="Intent" Status="Available"/>
        <ColorIntent ID="r0049" Class="Intent" Status="Available"/>
        <Component ID="r0047" Class="Quantity" Status="Available"
            ComponentType="Sheet" SpawnIDs="Sp0057"/>
    </ResourcePool>
    <AncestorPool>
        <Ancestor NodeID="SpawnTest" FileName="testjdf4.jdf"/>
    </AncestorPool>
</JDF>

```

Example O-5: 2-Component XJDF after Merging

In this example, it is assumed that the cover output component was created by some processor that processed the spawned Node. This resulted in the **Component** becoming available. The **Component** was also removed from the copy of the spawned Node, since it would otherwise exist twice.

TBD 2.x Example.

```

<JDF ID="SpawnTest" Type="Product" xmlns="http://www.CIP4.org/JDFSchema_1_1"
    Status="Waiting" Version="1.4" JobPartID="Part1">
    <AuditPool>
        <Created AgentName="CIP4 JDFWriter 1.0.1 beta"
            TimeStamp="2002-04-05T15:27:58+02:00"/>
        <Spawned URL="File:///spawn.jdf" jRef="n0046"
            TimeStamp="2002-04-05T15:34:43+02:00"
            NewSpawnID="Sp0057" rRefsRWCopied="r0047"/>
        <Merged URL="File:///spawn.jdf" jRef="n0046" MergeID="Sp0057"
            TimeStamp="2002-04-05T15:40:20+02:00" rRefsOverwritten="r0047"/>
    </AuditPool>
    <ResourcePool>
        <Component ID="r0043" Class="Quantity" Amount="10000"
            ComponentType="Sheet" Status="Unavailable"/>
        <BindingIntent ID="r0044" Class="Intent" Status="Available">
            <BindingType Actual="SoftCover" DataType="EnumerationSpan"/>
        </BindingIntent>
    </ResourcePool>
</JDF>

```

```

<ProductionIntent ID="r0045" Class="Intent" Status="Available">
    <PrintProcess Range="Gravure" DataType="EnumerationSpan"/>
</ProductionIntent>
<Component ID="r0047" Class="Quantity" ComponentType ="Sheet"
    Status="Available"/>
<Component ID="r0051" Class="Quantity" ComponentType="Sheet"
    Status="Unavailable"/>
</ResourcePool>
<ResourceLinkPool>
    <ComponentLink rRef="r0043" Usage="Output"/>
    <BindingIntentLink rRef="r0044" Usage="Input"/>
    <ProductionIntentLink rRef="r0045" Usage="Input"/>
    <ComponentLink rRef="r0047" Usage="Input"/>
    <ComponentLink rRef="r0051" Usage="Input"/>
</ResourceLinkPool>
<JDF ID="n0046" Type="Product" xmlns="http://www.CIP4.org/JDFSchema_1_1"
    Status="Waiting" Version="1.4" JobPartID="Part2"
    DescriptiveName="Cover">
    <AuditPool>
        <Created AgentName="CIP4 JDFWriter 1.0.1 beta"
           TimeStamp="2002-04-05T15:34:43+02:00"/>
    </AuditPool>
    <ResourceLinkPool>
        <ComponentLink rRef="r0047" Usage="Output"/>
        <LayoutIntentLink rRef="r0048" Usage="Input"/>
        <ColorIntentLink rRef="r0049" Usage="Input"/>
    </ResourceLinkPool>
    <ResourcePool>
        <LayoutIntent ID="r0048" Class="Intent" Status="Available"/>
        <ColorIntent ID="r0049" Class="Intent" Status="Available"/>
    </ResourcePool>
</JDF>
<JDF ID="n0050" Type="Product" Status="Waiting" JobPartID="Part3"
    DescriptiveName="Insert">
    <ResourceLinkPool>
        <ComponentLink rRef="r0051" Usage="Output"/>
        <LayoutIntentLink rRef="r0052" Usage="Input"/>
        <ColorIntentLink rRef="r0053" Usage="Input"/>
    </ResourceLinkPool>
    <ResourcePool>
        <LayoutIntent ID="r0052" Class="Intent" Status="Available"/>
        <ColorIntent ID="r0053" Class="Intent" Status="Available"/>
    </ResourcePool>
</JDF>
</JDF>

```

Example O-6: Partitioned ImageSetting Node before Spawning

The following example shows a simple **ImageSetting** Node that is Partitioned by *@Separation*. The Resources are not filled with data. The Input Resources are "Available".

TBD 2.x Example.

```

<JDF ID="n20020701190951" Type="ImageSetting"
    xmlns="http://www.CIP4.org/JDFSchema_1_1"
    Status="Waiting" JobPartID="ID777" Version="1.4">
    <ResourcePool>
        <ImageSetterParams ID="r0052" Class="Parameter" Locked="false"
            Status="Available"/>
        <Media ID="r0053" Class="Consumable" Locked="false" Status="Available"/>
    </ResourcePool>

```

```

<ExposedMedia ID="r0054" Class="Handling" Locked="false" Status="Unavailable"
    PartIDKeys="Separation">
    <MediaRef rRef="r0053"/>
    <ExposedMedia Separation="Cyan"/>
    <ExposedMedia Separation="Magenta"/>
    <ExposedMedia Separation="Yellow"/>
    <ExposedMedia Separation="Black"/>
</ExposedMedia>
<RunList ID="r0055" Class="Parameter" Locked="false" Status="Available"/>
</ResourcePool>
<ResourceLinkPool>
    <ImageSetterParamsLink rRef="r0052" Usage="Input"/>
    <MediaLink rRef="r0053" Usage="Input"/>
    <ExposedMediaLink rRef="r0054" Usage="Output"/>
    <RunListLink rRef="r0055" Usage="Input"/>
</ResourceLinkPool>
</JDF>

```

Example O-7: Spawning Cyan Partition of the ImageSetting Node

The following example shows the spawned Cyan Partition of the *ImageSetting* Node from the previous example.

```

<JDF ID="n20020701190951" Type="ImageSetting"
    xmlns="http://www.CIP4.org/JDFSchema_1_1" Status="Waiting" SpawnID="Sp0059"
    JobPartID="ID467" Version="1.4">
    <AuditPool/>
    <ResourcePool>
        <ImageSetterParams ID="r0052" Class="Parameter" Locked="true"
            Status="Available"/>
        <Media ID="r0053" Class="Consumable" Locked="true" Status="Available"
            PartIDKeys="Separation">
            <Media Separation="Cyan"/>
        </Media>
        <ExposedMedia ID="r0054" Class="Handling" Locked="true"
            Status="Unavailable" PartIDKeys="Separation">
            <ExposedMedia Separation="Cyan">
                <MediaRef rRef="r0053">
                    <Part Separation="Cyan"/>
                </MediaRef>
            </ExposedMedia>
        </ExposedMedia>
        <RunList ID="r0055" Class="Parameter" Locked="true" Status="Available"/>
    </ResourcePool>
    <ResourceLinkPool>
        <ImageSetterParamsLink rRef="r0052" Usage="Input"/>
        <MediaLink rRef="r0053" Usage="Input">
            <Part Separation="Cyan"/>
        </MediaLink>
        <ExposedMediaLink rRef="r0054" Usage="Output">
            <Part Separation="Cyan"/>
        </ExposedMediaLink>
        <RunListLink rRef="r0055" Usage="Input"/>
    </ResourceLinkPool>
    <AncestorPool>
        <Part Separation="Cyan"/>
        <Ancestor Type="ImageSetting" xmlns="http://www.CIP4.org/JDFSchema_1_1"
            NodeID="n20020701190951" Status="Waiting" Version="1.4"
            FileName="testjdf5.jdf"/>
    </AncestorPool>

```

```
</JDF>
```

Example O-8: Root Partitioned ImageSetting Node after Spawning

TBD 2.x Example.

```
<JDF ID="n20020701190951" Type="ImageSetting"
      xmlns="http://www.CIP4.org/JDFSchema_1_1" Status="Pool" JobPartID="ID778" Version="1.4">
  <AuditPool>
    <Spawned URL="File:///spawnIS.jdf" jRef="n20020701190951" Status="Waiting"
              TimeStamp="2002-07-01T19:18:03+02:00" NewSpawnID="Sp0059">
      <Part Separation="Cyan"/>
    </Spawned>
  </AuditPool>
  <ResourcePool>
    <NodeInfo Class="Parameter" ID="r050722154232_0045" NodeStatus="Waiting"
              PartIDKeys="Separation" Status="Available">
      <NodeInfo NodeStatus="Spawned" Separation="Cyan"/>
    </NodeInfo>
    <ImageSetterParams ID="r0052" Class="Parameter" Locked="false"
                       Status="Available" SpawnIDs="Sp0059" SpawnStatus="SpawnedRO" />
    <Media ID="r0053" Class="Consumable" Locked="false" Status="Available"
           SpawnIDs="Sp0059"/>
    <ExposedMedia ID="r0054" Class="Handling" Locked="false"
                  Status="Unavailable" SpawnIDs="Sp0059" PartIDKeys="Separation">
      <MediaRef rRef="r0053"/>
      <ExposedMedia Locked="true" Separation="Cyan" SpawnStatus="SpawnedRW" />
      <ExposedMedia Separation="Magenta"/>
      <ExposedMedia Separation="Yellow"/>
      <ExposedMedia Separation="Black"/>
    </ExposedMedia>
    <RunList ID="r0055" Class="Parameter" Locked="false" Status="Available"
              SpawnIDs="Sp0059" SpawnStatus="SpawnedRO" />
  </ResourcePool>
  <ResourceLinkPool>
    <ImageSetterParamsLink Usage="Input" rRef="r0052" />
    <MediaLink Usage="Input" rRef="r0053" />
    <ExposedMediaLink Usage="Output" rRef="r0054" />
    <RunListLink Usage="Input" rRef="r0055" />
    <NodeInfoLink Usage="Input" rRef="r050722154232_0045"/>
  </ResourceLinkPool>
</JDF>
```

Example O-9: Merged ImageSetting Node

The Node has now been executed and merged.

TBD 2.x Example.

```
<JDF ID="n20020701190951" Type="ImageSetting"
      xmlns="http://www.CIP4.org/JDFSchema_1_1" Status="Pool" JobPartID="ID234"
      Version="1.4">
  <AuditPool>
    <Spawned URL="File:///spawnIS.jdf" jRef="n20020701190951"
              Status="Waiting"
              TimeStamp="2002-07-01T20:25:03+02:00" NewSpawnID="Sp0059">
      <Part Separation="Cyan"/>
    </Spawned>
  </AuditPool>
```

```

</Spawned>
<Merged URL="File:///spawnIS2.jdf" jRef="n20020701190951"
    MergeID="Sp0059"
    TimeStamp="2002-07-01T20:27:51+02:00">
    <Part Separation="Cyan"/>
</Merged>
</AuditPool>
<ResourcePool>
    <ImageSetterParams ID="r0052" Class="Parameter" Status="Available"/>
    <Media ID="r0053" Class="Consumable" Status="Available"/>
    <ExposedMedia ID="r0054" Class="Handling" Status="Unavailable"
        PartIDKeys="Separation">
        <MediaRef rRef="r0053"/>
        <ExposedMedia Status="Available" Separation="Cyan"/>
        <ExposedMedia Separation="Magenta"/>
        <ExposedMedia Separation="Yellow"/>
        <ExposedMedia Separation="Black"/>
    </ExposedMedia>
    <RunList ID="r0055" Class="Parameter" Status="Available"/>
    <NodeInfo Class="Parameter" ID="r050722154232_0045" NodeStatus="Waiting"
        PartIDKeys="Separation" Status="Available">
        <NodeInfo NodeStatus="Completed" Separation="Cyan"/>
    </NodeInfo>
</ResourcePool>
<ResourceLinkPool>
    <ImageSetterParamsLink rRef="r0052" Usage="Input"/>
    <MediaLink rRef="r0053" Usage="Input"/>
    <ExposedMediaLink rRef="r0054" Usage="Output"/>
    <RunListLink rRef="r0055" Usage="Input"/>
    <NodeInfoLink Usage="Input" rRef="r050722154232_0045"/>
</ResourceLinkPool>
</JDF>

```

O.4 RunList

Example O-10: RunList

The following example shows the various separation types, all mixed into one big **RunList**. Both in-line and *@IDREF* versions of **RunList** are used.

TBD 2.x Example.

```

<RunList Class="Parameter" ID="Link0003" NPage="10"
    PartIDKeys="Run Separation"
    Status="Available">
    <Comment>Preseparated Runs in multiple files
        All LayoutElements are inline resources
    </Comment>
    <RunList FirstPage="0" NPage="1" Run="1">
        <RunList Separation="Cyan">
            <LayoutElement>
                <FileSpec URL="File:///Cyan.pdf"/>
            </LayoutElement>
        </RunList>
        <RunList Separation="Magenta">
            <LayoutElement>
                <FileSpec URL="File:///Magenta.pdf"/>
            </LayoutElement>
        </RunList>
    </RunList>

```

```

<RunList Separation="Yellow">
    <LayoutElement>
        <FileSpec URL="File:///Yellow.pdf"/>
    </LayoutElement>
</RunList>
<RunList Separation="Black">
    <LayoutElement>
        <FileSpec URL="File:///Black.pdf"/>
    </LayoutElement>
</RunList>
<RunList Separation="SpotGreen">
    <LayoutElement>
        <FileSpec URL="File:///Green.pdf"/>
    </LayoutElement>
</RunList>
</RunList>
<RunList NPage="2" Run="2" SkipPage="4">
    <Comment>
        Preseparated Runs in one file CMYKGCMYKG
        LayoutElements are inter-resource links
    </Comment>
    <RunList FirstPage="0" Separation="Cyan">
        <LayoutElementRef rRef="Link0004"/>
    </RunList>
    <RunList FirstPage="1" Separation="Magenta">
        <LayoutElementRef rRef="Link0004"/>
    </RunList>
    <RunList FirstPage="2" Separation="Yellow">
        <LayoutElementRef rRef="Link0004"/>
    </RunList>
    <RunList FirstPage="3" Separation="Black">
        <LayoutElementRef rRef="Link0004"/>
    </RunList>
    <RunList FirstPage="4" Separation="SpotGreen">
        <LayoutElementRef rRef="Link0004"/>
    </RunList>
</RunList>
<RunList NPage="1" Run="3" SkipPage="3">
    <Comment>
        No Magenta, the missing sep does not exist as a page
    </Comment>
    <RunList FirstPage="10" Separation="Cyan">
        <LayoutElementRef rRef="Link0004"/>
    </RunList>
    <RunList FirstPage="11" Separation="Yellow">
        <LayoutElementRef rRef="Link0004"/>
    </RunList>
    <RunList FirstPage="12" Separation="Black">
        <LayoutElementRef rRef="Link0004"/>
    </RunList>
    <RunList FirstPage="13" Separation="Green">
        <LayoutElementRef rRef="Link0004"/>
    </RunList>
</RunList>
<RunList NPage="2" Run="4" SkipPage="4">
    <Comment>
        Continuation of Preseparated Runs in one file CMYKGCMYKG -
        the missing sep of the previous page does not exist as a page
    </Comment>

```

```

</Comment>
<RunList FirstPage="14" Separation="Cyan">
    <LayoutElementRef rRef="Link0004"/>
</RunList>
<RunList FirstPage="15" Separation="Magenta">
    <LayoutElementRef rRef="Link0004"/>
</RunList>
<RunList FirstPage="16" Separation="Yellow">
    <LayoutElementRef rRef="Link0004"/>
</RunList>
<RunList FirstPage="17" Separation="Black">
    <LayoutElementRef rRef="Link0004"/>
</RunList>
<RunList FirstPage="18" Separation="SpotGreen">
    <LayoutElementRef rRef="Link0004"/>
</RunList>
</RunList>
<RunList NPage="2" Run="5">
    <Comment>
        Preseparated Runs in one file CCMMYYKKGG
    </Comment>
    <RunList FirstPage="0" Separation="Cyan">
        <LayoutElementRef rRef="Link0005"/>
    </RunList>
    <RunList FirstPage="2" Separation="Magenta">
        <LayoutElementRef rRef="Link0005"/>
    </RunList>
    <RunList FirstPage="4" Separation="Yellow">
        <LayoutElementRef rRef="Link0005"/>
    </RunList>
    <RunList FirstPage="6" Separation="Black">
        <LayoutElementRef rRef="Link0005"/>
    </RunList>
    <RunList FirstPage="8" Separation="SpotGreen">
        <LayoutElementRef rRef="Link0005"/>
    </RunList>
</RunList>
<RunList NPage="2" Run="6">
    <Comment>
        Combined Runs in one file
    </Comment>
    <LayoutElement ElementType="Document">
        <FileSpec URL="File:///Combined.pdf"/>
    </LayoutElement>
</RunList>
</RunList>
<LayoutElement Class="Parameter" ID="Link0004" Status="Available">
    <FileSpec URL="File:///PreSepCMYKG.pdf"/>
</LayoutElement>
<LayoutElement Class="Parameter" ID="Link0005" Status="Available">
    <FileSpec URL="File:///PreSepCCMMYYKKGG.pdf"/>
</LayoutElement>

```

O.5 Messages

O.5.1 Simple KnownMessages

The following simple example shows a KnownMessages Query Message and the Response Message sent by a fairly dumb Controller:

Example O-11: KnownMessages Query

TBD 2.x Example.

Example O-12: KnownMessages Response

TBD 2.x Example.

O.5.2 Simple persistent channel

The following query requests a persistent channel for Status Messages. An update is requested whenever an Attribute changes. Then the following four examples are a set of typical, simple responses that are emitted whenever *@DeviceStatus* changes; three responses are Signal Messages

Example O-13: Status Query

TBD 2.x Example.

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="JMFCClient"
     TimeStamp="2000-11-07T16:02:09+01:00" MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<Query ID="Q0011" Type="Status" xsi:type="QueryStatus">
    <Subscription URL="http://123.123.123.123/message/recipient">
        <ObservationTarget ObservationPath="//*/@*"/>
    </Subscription>
    <StatusQuParams JobDetails="Brief"/>
</Query>
</JMF>
```

Example O-14: Status Response

This is the Response Message that is sent immediately within the same HTTP connection as the Query Message.

TBD 2.x Example.

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="JMFCClient #2"
     TimeStamp="2000-11-07T16:02:19+01:00"
      MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<Response ID="R0013" Type="Status" xsi:type="ResponseStatus" refID="Q0011">
    <DeviceInfo DeviceStatus="Idle"/>
</Response>
</JMF>
```

Example O-15: Status Signal #1

This is an intermediate Signal that was emitted when *@DeviceStatus* changed.

TBD 2.x Example.

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="JMFCClient #2"
     TimeStamp="2000-11-07T17:02:19+01:00" MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<Signal ID="Q0015" Type="Status" xsi:type="SignalStatus" refID="Q0011">
```

```

<DeviceInfo DeviceStatus="Setup"/>
</Signal>
</JMF>

```

Example O-16: Status Signal #2

This is an intermediate Signal that was emitted when *@DeviceStatus* changed.

TBD 2.x Example.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="JMFCClient #2"
     TimeStamp="2000-11-07T17:08:19+01:00" MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
  <Signal ID="Q0017" Type="Status" xsi:type="SignalStatus" refID="Q0011">
    <DeviceInfo DeviceStatus="Running"/>
  </Signal>
</JMF>

```

Example O-17: Status Signal #3

This is the last Signal of the persistent channel.

TBD 2.x Example.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="JMFCClient #2"
     TimeStamp="2000-11-07T19:02:19+01:00" MaxVersion="1.4" Version="1.4"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
  <Signal ID="Q0017" Type="Status" xsi:type="SignalStatus" refID="Q0011"
          LastRepeat="true">
    <DeviceInfo DeviceStatus="Idle"/>
  </Signal>
</JMF>

```

O.5.3 XJMF Pipe Messages

The following example details the sequence of XJMF pipe messages from a digital printer to a connected finisher. In this example, we send one *@Operation = Push* XJMF per printed sheet. the XJDF files are skeletons with little detail.

O.5.3.1 Example Printer XJDF

XJDF for dynamic pipes at the printer:

```

<JDF ID="n_000004" JobID="J1" JobPartID="n_000002.1" MaxVersion="1.4"
      Status="Waiting" Type="Combined" Types="DigitalPrinting" Version="1.4"
      xmlns="http://www.CIP4.org/JDFSchema_1_1"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Combined">
  <AuditPool>
    <Created AgentName="CIP4 JDF Writer Java" AgentVersion="1.4a BLD 74"
              ID="a_000005" TimeStamp="2013-09-19T15:36:28+02:00"/>
  </AuditPool>
  <ResourcePool>
    <RunList Automation="Dynamic" Class="Parameter" ID="r_000006"
              Status="Available"/>
    <DigitalPrintingParams Class="Parameter" ID="r_000007"
              Status="Available"/>
    <Component Automation="Dynamic" Class="Quantity"
              ComponentType="PartialProduct Sheet" ID="r_000008"
              PartIDKeys="SetIndex DocTags" PipeID="PipeSheet"
              PipeProtocol="JMFPush" Status="Unavailable">
      <Component SetIndex="0~1">
        <Component DocTags="Cover" SurfaceCount="2"/>
      </Component>
    </Component>
  </ResourcePool>
</JDF>

```

```

        <Component DocTags="Body" SurfaceCount="-1"/>
    </Component>
</Component>
</ResourcePool>
<ResourceLinkPool>
    <RunListLink CombinedProcessIndex="0" Usage="Input" rRef="r_000006"/>
    <DigitalPrintingParamsLink CombinedProcessIndex="0" Usage="Input"
        rRef="r_000007"/>
    <ComponentLink Amount="1" CombinedProcessIndex="0"
        Usage="Output" rRef="r_000008">
        <Part SetIndex="0~1"/>
    </ComponentLink>
</ResourceLinkPool>
</JDF>

```

O.5.3.2 Example Finisher XJDF

XJDF for dynamic pipes at the finisher:

```

<JDF ID="n_000009" JobID="J1" JobPartID="n_000002.2" MaxVersion="1.4"
    Status="Waiting" Type="Combined" Types="Collecting Stitching"
    Version="1.4" xmlns="http://www.CIP4.org/JDFSchema_1_1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Combined">
    <AuditPool>
        <Created AgentName="CIP4 JDF Writer Java" AgentVersion="1.4a BLD 74"
            ID="a_000010" TimeStamp="2013-09-19T15:36:28+02:00"/>
    </AuditPool>
    <ResourceLinkPool>
        <ComponentLink ProcessUsage="Cover" Usage="Input" rRef="r_000008">
            <Part DocTags="Cover" SetIndex="0~1"/>
        </ComponentLink>
        <ComponentLink CombinedProcessIndex="0" Usage="Input" rRef="r_000008">
            <Part DocTags="Body" SetIndex="0~1"/>
        </ComponentLink>
        <StitchingParamsLink CombinedProcessIndex="1" Usage="Input"
            rRef="r_000011"/>
        <ComponentLink Amount="1" CombinedProcessIndex="1"
            Usage="Output" rRef="r_000012"/>
    </ResourceLinkPool>
    <ResourcePool>
        <StitchingParams Class="Parameter" ID="r_000011" Status="Available"/>
        <Component Class="Quantity" ComponentType="FinalProduct Block"
            ID="r_000012" Status="Unavailable"/>
        <Component Automation="Dynamic" Class="Quantity"
            ComponentType="PartialProduct Sheet" ID="r_000008"
            PartIDKeys="SetIndex DocTags" PipeID="PipeSheet"
            PipeProtocol="JMFPush" Status="Unavailable">
            <Component SetIndex="0~1">
                <Component DocTags="Cover" SurfaceCount="2"/>
                <Component DocTags="Body" SurfaceCount="-1"/>
            </Component>
        </Component>
    </ResourcePool>
</JDF>

```

O.5.3.3 Initiation of PipePush sequence

The communication is initiated with a PipePush for the cover sheet of the first set.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:26+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
  <Command ID="m.1349935098._000002" Type="PipePush"
            xsi:type="CommandPipePush">
    <PipeParams JobID="J1" PipeID="PipeSheet">
      <AmountPool>
        <PartAmount Amount="1">
          <Part DocTags="Cover" SetIndex="0"/>
        </PartAmount>
        <PartAmount ActualAmount="1">
          <Part DocTags="Cover" SetIndex="0" SheetIndex="0"/>
        </PartAmount>
      </AmountPool>
    </PipeParams>
  </Command>
</JMF>

```

O.5.3.4 Continuation of PipePush sequence

The communication continues with the @Operation = Push messages for the 5 body sheets of the first set, and the cover and body of the following sets.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
  <Command ID="m.1349935098._000003" Type="PipePush"
            xsi:type="CommandPipePush">
    <PipeParams JobID="J1" PipeID="PipeSheet">
      <AmountPool>
        <PartAmount Amount="5">
          <Part DocTags="Body" SetIndex="0"/>
        </PartAmount>
        <PartAmount ActualAmount="1">
          <Part DocTags="Body" SetIndex="0" SheetIndex="0"/>
        </PartAmount>
      </AmountPool>
    </PipeParams>
  </Command>
</JMF>

```

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
  <Command ID="m.1349935098._000004" Type="PipePush"
            xsi:type="CommandPipePush">
    <PipeParams JobID="J1" PipeID="PipeSheet">
      <AmountPool>
        <PartAmount Amount="5">
          <Part DocTags="Body" SetIndex="0"/>
        </PartAmount>
        <PartAmount ActualAmount="1">
          <Part DocTags="Body" SetIndex="0" SheetIndex="1"/>
        </PartAmount>
      </AmountPool>
    </PipeParams>
  </Command>
</JMF>

```

Appendix O Examples

```

        </PipeParams>
    </Command>
</JMF>
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000005" Type="PipePush">
    <xsi:type>CommandPipePush</xsi:type>
    <PipeParams JobID="J1" PipeID="PipeSheet">
        <AmountPool>
            <PartAmount Amount="5">
                <Part DocTags="Body" SetIndex="0"/>
            </PartAmount>
            <PartAmount ActualAmount="1">
                <Part DocTags="Body" SetIndex="0" SheetIndex="4"/>
            </PartAmount>
        </AmountPool>
    </PipeParams>
</Command>
</JMF>
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000006" Type="PipePush">
    <xsi:type>CommandPipePush</xsi:type>
    <PipeParams JobID="J1" PipeID="PipeSheet">
        <AmountPool>
            <PartAmount Amount="1">
                <Part DocTags="Cover" SetIndex="1"/>
            </PartAmount>
            <PartAmount ActualAmount="1">
                <Part DocTags="Cover" SetIndex="1" SheetIndex="0"/>
            </PartAmount>
        </AmountPool>
    </PipeParams>
</Command>
</JMF>
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000007" Type="PipePush">
    <xsi:type>CommandPipePush</xsi:type>
    <PipeParams JobID="J1" PipeID="PipeSheet">
        <AmountPool>
            <PartAmount Amount="1">
                <Part DocTags="Body" SetIndex="1"/>
            </PartAmount>
            <PartAmount ActualAmount="1">
                <Part DocTags="Body" SetIndex="1" SheetIndex="0"/>
            </PartAmount>
        </AmountPool>
    </PipeParams>
</Command>
</JMF>

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
```

```

<xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000008" Type="PipePush"
    xsi:type="CommandPipePush">
<PipeParams JobID="J1" PipeID="PipeSheet">
    <AmountPool>
        <PartAmount Amount="7">
            <Part DocTags="Body" SetIndex="35"/>
        </PartAmount>
        <PartAmount ActualAmount="1">
            <Part DocTags="Body" SetIndex="35" SheetIndex="4"/>
        </PartAmount>
    </AmountPool>
</PipeParams>
</Command>
</JMF>

```

O.5.3.5 Paper Jam in finisher - PipePause

Due to a paper jam that destroys sets 34 and 35, a @Operation = Pause is sent from finisher to printer. Note that specifying which sheets were destroyed is optional at this point.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
    SenderID="Finisher"TimeStamp="2013-09-18T10:58:27+02:00"
    Version="1.5" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="JMFRootMessage">
<Command ID="m.609016672._000009" Type="PipePause"
    xsi:type="CommandPipePause">
<PipeParams JobID="J1" PipeID="PipeSheet">
    <AmountPool>
        <PartAmount>
            <Part Condition="Waste" SetIndex="34 35"/>
        </PartAmount>
    </AmountPool>
</PipeParams>
</Command>
</JMF>

```

O.5.3.6 Paper jam cleanup in finisher - PipePull

After the paper jam has been cleaned and the finisher is ready to receive sheets, it sends a @Operation = Pull specifying which sets should be produced.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
    SenderID="TestSender"TimeStamp="2013-09-18T10:58:27+02:00"
    Version="1.5" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="JMFRootMessage">
<Command ID="m.587730553._000010" SenderID="Finisher" Type="PipePull"
    xsi:type="CommandPipePull">
<PipeParams JobID="J1" PipeID="PipeSheet">
    <AmountPool>
        <PartAmount>
            <Part SetIndex="34~1"/>
        </PartAmount>
    </AmountPool>
</PipeParams>
</Command>
</JMF>

```

O.5.3.7 Continued PipePush

The printer continues by sending sheets starting at the point requested by the previous @Operation = Pull.

```

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000011" Type="PipePush"
          xsi:type="CommandPipePush">
<PipeParams JobID="J1" PipeID="PipeSheet">
<AmountPool>
    <PartAmount Amount="1">
        <Part DocTags="Cover" SetIndex="34"/>
    </PartAmount>
    <PartAmount ActualAmount="1">
        <Part DocTags="Cover" SetIndex="34" SheetIndex="0"/>
    </PartAmount>
</AmountPool>
</PipeParams>
</Command>
</JMF>

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000012" Type="PipePush"
          xsi:type="CommandPipePush">
<PipeParams JobID="J1" PipeID="PipeSheet">
<AmountPool>
    <PartAmount Amount="5">
        <Part DocTags="Body" SetIndex="34"/>
    </PartAmount>
    <PartAmount ActualAmount="1">
        <Part DocTags="Body" SetIndex="34" SheetIndex="0"/>
    </PartAmount>
</AmountPool>
</PipeParams>
</Command>
</JMF>

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000013" Type="PipePush"
          xsi:type="CommandPipePush">
<PipeParams JobID="J1" PipeID="PipeSheet">
<AmountPool>
    <PartAmount Amount="7">
        <Part DocTags="Body" SetIndex="122"/>
    </PartAmount>
    <PartAmount ActualAmount="1">
        <Part DocTags="Body" SetIndex="122" SheetIndex="4"/>
    </PartAmount>
</AmountPool>
</PipeParams>
</Command>
</JMF>
```

O.5.3.8 Paper jam in printer - PipePause

We now have a malfunction in the printer, that makes set 122 unusable. Therefore the printer sends a @Operation = Pause to the finisher specifying which sheets are waste.

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000014" Type="PipePause"
          xsi:type="CommandPipePause">
<PipeParams JobID="J1" PipeID="PipeSheet">
<AmountPool>
<PartAmount>
<Part Condition="Waste" SetIndex="122"/>
</PartAmount>
</AmountPool>
</PipeParams>
</Command>
</JMF>
```

O.5.3.9 Optional PipePause from the Finisher

The finisher MAY pause the pipe in case a pause from the Producer causes issues in the Finisher

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Finisher"TimeStamp="2013-10-16T15:04:30+02:00"
      Version="1.5" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="JMFRootMessage">
<Command ID="m.609016672._000015" Type="PipePause"
          xsi:type="CommandPipePause">
<PipeParams JobID="J1" PipeID="PipeSheet">
<AmountPool>
<PartAmount>
<Part Condition="Waste" SetIndex="122"/>
</PartAmount>
</AmountPool>
</PipeParams>
</Command>
</JMF>
```

O.5.3.10 Optional PipePull from the Finisher

The printer continues by sending sheets starting at the point requested by the previous @Operation = Pull.

PipePullFinisher2Example.jmf

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="TestSender"TimeStamp="2013-10-16T15:04:30+02:00"
      Version="1.5" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="JMFRootMessage">
<Command ID="m.587730553._000016" SenderID="Finisher" Type="PipePull"
          xsi:type="CommandPipePull">
<PipeParams JobID="J1" PipeID="PipeSheet">
<AmountPool>
<PartAmount>
<Part SetIndex="122~-1"/>
</PartAmount>
</AmountPool>
</PipeParams>
</Command>
</JMF>
```

O.5.3.11 Paper jam cleanup in printer - PipePush

After cleanup, the printer continues with the first sheet of the destroyed set and sends @Operation = Push until it has completed.

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000015" Type="PipePush"
          xsi:type="CommandPipePush">
    <PipeParams JobID="J1" PipeID="PipeSheet">
        <AmountPool>
            <PartAmount Amount="1">
                <Part DocTags="Cover" SetIndex="122"/>
            </PartAmount>
            <PartAmount ActualAmount="1">
                <Part DocTags="Cover" SetIndex="122" SheetIndex="0"/>
            </PartAmount>
        </AmountPool>
    </PipeParams>
</Command>
</JMF>

<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000016" Type="PipePush"
          xsi:type="CommandPipePush">
    <PipeParams JobID="J1" PipeID="PipeSheet">
        <AmountPool>
            <PartAmount Amount="6">
                <Part DocTags="Body" SetIndex="221"/>
            </PartAmount>
            <PartAmount ActualAmount="1">
                <Part DocTags="Body" SetIndex="221" SheetIndex="5"/>
            </PartAmount>
        </AmountPool>
    </PipeParams>
</Command>
</JMF>
```

O.5.3.12 Job done - PipeClose

The printer is now finished and sends a @Operation = Close with a summary of what it believes to have sent.

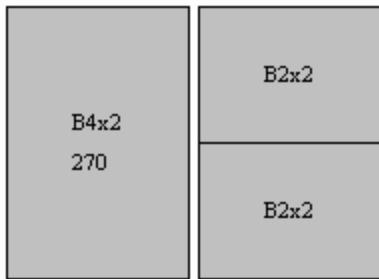
```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" MaxVersion="1.5"
      SenderID="Printer"TimeStamp="2013-09-18T10:58:27+02:00" Version="1.5"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="JMFRootMessage">
<Command ID="m.1349935098._000017" Type="PipeClose"
          xsi:type="CommandPipeClose">
    <PipeParams JobID="J1" PipeID="PipeSheet">
        <AmountPool>
            <PartAmount>
                <Part SetIndex="0~221"/>
            </PartAmount>
        </AmountPool>
    </PipeParams>
</Command>
```

</ JMF >

O.6 Stripping

Example O-18: Using Position

The following example illustrates the more advanced use of the `@Position` object. Note that the two B2x2 Signatures are filled independently.



TBD 2.x Example.

```

<BinderySignature Class="Parameter" ID="B4x2" NumberUp="4 2"
    Status="Available"/>
<BinderySignature Class="Parameter" ID="B2x2" NumberUp="2 2"
    Status="Available"/>
<StrippingParams Class="Parameter" ID="L1"
    PartIDKeys="SheetName BinderySignatureName"
    Status="Available" WorkStyle="WorkAndBack">
    <StrippingParams SheetName="Sheet1">
        <StrippingParams BinderySignatureName="B4x2">
            <BinderySignatureRef rRef="B4x2"/>
            <Position RelativeBox="0 0 0.5 1" Orientation="Rotate270"/>
        </StrippingParams>
        <StrippingParams BinderySignatureName="B2x2-1">
            <BinderySignatureRef rRef="B2x2"/>
            <Position RelativeBox="0.5 0 1 0.5"/>
        </StrippingParams>
        <StrippingParams BinderySignatureName="B2x2-2">
            <BinderySignatureRef rRef="B2x2"/>
            <Position RelativeBox="0.5 0.5 1 1"/>
        </StrippingParams>
    </StrippingParams>
</StrippingParams>

```

Example O-19: Multiple Bindery Signatures

The following example illustrates how two identical `BinderySignature` Resources that represent the same sections are placed onto a surface. It also shows how `SignatureCell` elements are overwritten for various sections.

A3	A0	Section A
B3	B0	Section B

TBD 2.x Example.

```

<BinderySignature Class="Parameter" ID="B4x2" NumberUp="4 2"
    Status="Available"/>
<BinderySignature Class="Parameter" ID="B2x2" NumberUp="2 2"
    Status="Available"/>
<StrippingParams Class="Parameter" ID="L1" PartUsage="Implicit"
    Status="Available"
    PartIDKeys="SheetName BinderySignatureName CellIndex"
    WorkStyle="WorkAndBack">
    <BinderySignatureRef rRef="B4x2"/>
    <StrippingParams JobID="Customer Job 1" SheetName="Sheet1">
        <StrippingParams BinderySignatureName="B4x2">
            <BinderySignatureRef rRef="B4x2"/>
            <StripCellParams BleedFace="42" BleedSpine="0" MillingDepth="21"/>
            <Position RelativeBox="0 0 0.5 1" Orientation="Rotate270"/>
        </StrippingParams>
        <StrippingParams BinderySignatureName="B2x2">
            <BinderySignatureRef rRef="B2x2"/>
            <StripCellParams BleedFace="42" BleedSpine="20"
                MillingDepth="84"/>
            <Position RelativeBox="0.5 0 1 0.5"/>
            <Position RelativeBox="0.5 0.5 1 1"/>
            <StrippingParams CellIndex="3">
                <StripCellParams BleedFace="10" BleedSpine="10"
                    MillingDepth="48"/>
            </StrippingParams>
        </StrippingParams>
    </StrippingParams>
</StrippingParams>
</StrippingParams>
</StrippingParams>
</StrippingParams>

```

Example O-20: Multisection Bindery Signatures

The following example illustrates the imposition of a Job containing 80 pages using *ComeAndGo*. Five Sheets need to be produced each containing two sections.

		B8	B1	B4	B5
A4	A5	A8	A1		

TBD 2.x Example.

```

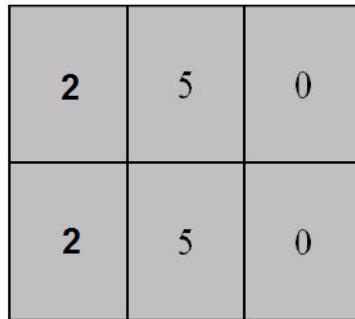
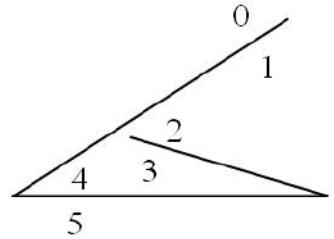
<BinderySignature Class="Parameter" ID="ComeAndGo" NumberUp="4 2"
    Status="Available">
    <SignatureCell BackPages="1 6 5 2" FrontPages="3 4 7 0" Orientation="Up"
        SectionIndex="0"/>
    <SignatureCell BackPages="6 1 2 5" FrontPages="4 3 0 7"
        Orientation="Down" SectionIndex="1"/>
</BinderySignature>
<StrippingParams Class="Parameter" ID="L1" PartIDKeys="SheetName"
    Status="Available" WorkStyle="WorkAndBack">
    <BinderySignatureRef rRef="ComeAndGo"/>
    <StrippingParams SectionList="0 9" SheetName="Sheet1"/>
    <StrippingParams SectionList="1 8" SheetName="Sheet2"/>
    <StrippingParams SectionList="2 7" SheetName="Sheet3"/>
    <StrippingParams SectionList="3 6" SheetName="Sheet4"/>
    <StrippingParams SectionList="4 5" SheetName="Sheet5"/>
</StrippingParams>

<BinderySignature Class="Parameter" ID="ComeAndGo" NumberUp="4 2"
    Status="Available">
    <SignatureCell BackPages="1 6 5 2" FrontPages="3 4 7 0" Orientation="Up"
        SectionIndex="0"/>
    <SignatureCell BackPages="6 1 2 5" FrontPages="4 3 0 7"
        Orientation="Down" SectionIndex="1"/>
</BinderySignature>
<StrippingParams Class="Parameter" ID="L1" JobID="MyJob"
    PartIDKeys="SheetName SectionIndex" Status="Available"
    WorkStyle="WorkAndBack">
    <BinderySignatureRef rRef="ComeAndGo"/>
    <StrippingParams SheetName="Sheet1">
        <StrippingParams AssemblyIDs="Book1" SectionIndex="0" SectionList="0"/>
        <StrippingParams AssemblyIDs="Book2" SectionIndex="1" SectionList="9"/>
    </StrippingParams>
    <StrippingParams SheetName="Sheet2">
        <StrippingParams AssemblyIDs="Book1" SectionIndex="0" SectionList="1"/>
        <StrippingParams AssemblyIDs="Book2" SectionIndex="1" SectionList="8"/>
    </StrippingParams>
    <StrippingParams SheetName="Sheet3">
        <StrippingParams AssemblyIDs="Book1" SectionIndex="0" SectionList="2"/>
        <StrippingParams AssemblyIDs="Book2" SectionIndex="1" SectionList="7"/>
    </StrippingParams>
    <StrippingParams SheetName="Sheet4">
        <StrippingParams AssemblyIDs="Book1" SectionIndex="0" SectionList="3"/>
        <StrippingParams AssemblyIDs="Book2" SectionIndex="1" SectionList="6"/>
    </StrippingParams>
    <StrippingParams SheetName="Sheet5">
        <StrippingParams AssemblyIDs="Book1" SectionIndex="0" SectionList="4"/>
        <StrippingParams AssemblyIDs="Book2" SectionIndex="1" SectionList="5"/>
    </StrippingParams>
</StrippingParams>

```

Example O-21: FoldOuts

The following example illustrates the use of foldouts. The same foldout is placed twice on a press Sheet.



TBD 2.x Example.

```

<BinderySignature Class="Parameter" ID="foldout" NumberUp="2 1"
    Status="Available">
    <!-foldout cell-->
    <SignatureCell FrontPages="2" BackPages="3" Orientation="Up"/>
    <!- back page cell which has fold out attached -->
    <SignatureCell FrontPages="4" BackPages="5" FaceCells="0"
        Orientation="Up"/>
    <!- front page cell -->
    <SignatureCell FrontPages="0" BackPages="1" Orientation="Up"/>
</BinderySignature>
<StrippingParams Class="Parameter" ID="Cover" Status="Available"
    WorkStyle="WorkAndBack">
    <BinderySignatureRef rRef="foldout"/>
    <Position RelativeBox="0 0 1 0.5"/>
    <Position RelativeBox="0 0.5 1 1"/>
</StrippingParams>
```

Example O-22: Multiple Web Layout

The following example illustrates a regular double-Web layout. A double-Web **BinderySignature** is used in two Signatures. This results in four Sheets.

			23	8	15	16
31	0	7	24			

Web1

		21	10	13	18
29	2	5	26		

Web2

TBD 2.x Example.

```

<BinderySignature Class="Parameter" ID="B001" NumberUp="4 2"
  PartIDKeys="WebName" Status="Available">
  <BinderySignature WebName="Web1">
    <SignatureCell BackPages="22 9 14 17" FrontPages="31 0 7 24"
      Orientation="Up"/>
    <SignatureCell BackPages="25 6 1 30" FrontPages="16 15 8 23"
      Orientation="Down"/>
  </BinderySignature>
  <BinderySignature WebName="Web2">
    <SignatureCell BackPages="20 11 12 19" FrontPages="29 2 5 26"
      Orientation="Up"/>
    <SignatureCell BackPages="27 4 3 28" FrontPages="18 13 10 21"
      Orientation="Down"/>
  </BinderySignature>
</BinderySignature>
<StrippingParams Class="Parameter" ID="MultiWeb1"
  PartIDKeys="SignatureName SheetName"
  Status="Available" WorkStyle="WorkAndBack">
  <StrippingParams SignatureName="Signature1">
    <StrippingParams SheetName="Sheet1">
      <BinderySignatureRef rRef="B001">
        <Part WebName="Web1"/>
      </BinderySignatureRef>
    </StrippingParams>
    <StrippingParams SheetName="Sheet2">
      <BinderySignatureRef rRef="B001">
        <Part WebName="Web2"/>
      </BinderySignatureRef>
    </StrippingParams>
  </StrippingParams>
  <StrippingParams SignatureName="Signature2">
    <StrippingParams SheetName="Sheet3">
      <BinderySignatureRef rRef="B001">

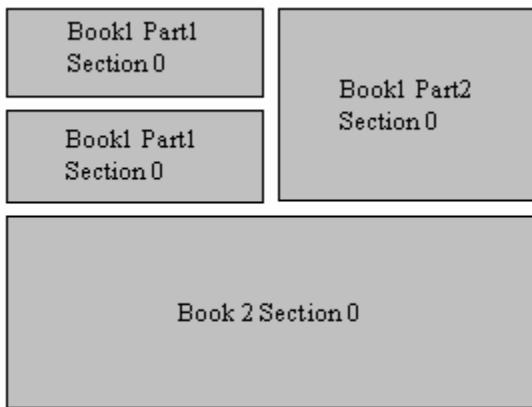
```

```
<Part WebName="Web1"/>
</BinderySignatureRef>
</StrippingParams>
<StrippingParams SheetName="Sheet4">
    <BinderySignatureRef rRef="B001">
        <Part WebName="Web2"/>
    </BinderySignatureRef>
</StrippingParams>
</StrippingParams>
</StrippingParams>
```

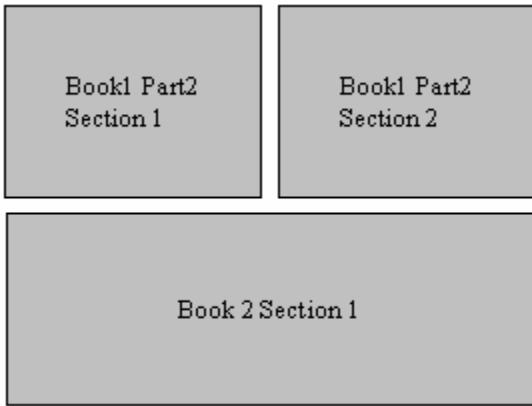
Example O-23: Stripping Process

The next sample illustrates the **Stripping** Process and its **StrippingParams** and **Assembly** Resources.

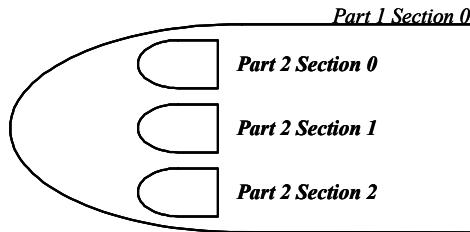
Sheet1:



Sheet2:



Assembly 1



TBD 2.x Example.

```

<JDF ID="n001" Type="Stripping" JobPartID="ID378" Status="Ready" Version="1.4">
  <ResourcePool>
    <BinderySignature Class="Parameter" FoldCatalog="F4-1" ID="F4-1"
      Status="Available"/>
    <BinderySignature Class="Parameter" FoldCatalog="F16-6" ID="F16-6"
      Status="Available"/>
    <BinderySignature Class="Parameter" FoldCatalog="F8-7" ID="F8-7"
      Status="Available"/>
    <StrippingParams Class="Parameter" ID="L1"
      PartIDKeys="SheetName BinderySignatureName" Status="Available">
      <StrippingParams SheetName="Sheet1">
        <StrippingParams AssemblyIDs="Part1" BinderySignatureName="F4-1"
          JobID="Book1" SectionList="0">
          <BinderySignatureRef rRef="F4-1"/>
          <Position RelativeBox="0 0.5 0.5 0.75"/>
          <Position RelativeBox="0 0.75 0.5 1"/>
        </StrippingParams>
        <StrippingParams AssemblyIDs="Part2" BinderySignatureName="F8-7"
          JobID="Book1" SectionList="0">
          <BinderySignatureRef rRef="F8-7"/>
          <Position RelativeBox="0.5 0.5 1 1"/>
        </StrippingParams>
        <StrippingParams BinderySignatureName="F16-6" JobID="Book2"
          SectionList="0">
          <BinderySignatureRef rRef="F16-6"/>
          <Position RelativeBox="0 0 1 0.5"/>
        </StrippingParams>
      </StrippingParams>
      <StrippingParams SheetName="Sheet2">
        <StrippingParams AssemblyIDs="Part2" BinderySignatureName="F8-7_1"
          JobID="Book1" SectionList="1">
          <BinderySignatureRef rRef="F8-7"/>
          <Position RelativeBox="0 0.5 0.5 1"/>
        </StrippingParams>
        <StrippingParams AssemblyIDs="Part2" BinderySignatureName="F8-7_2"
          JobID="Book1" SectionList="2">
          <BinderySignatureRef rRef="F8-7"/>
          <Position RelativeBox="0.5 0.5 1 1"/>
        </StrippingParams>
        <StrippingParams BinderySignatureName="F16-6" JobID="Book2"
          SectionList="1">
          <BinderySignatureRef rRef="F16-6"/>
        </StrippingParams>
      </StrippingParams>
    </ResourcePool>
  </JDF>

```

```

        <Position RelativeBox="0 0 1 0.5"/>
    </StrippingParams>
</StrippingParams>
</StrippingParams>
<Assembly Class="Parameter" ID="A1" JobID="Book1" Order="List"
    Status="Available">
    <AssemblySection AssemblyIDs="Part1" Order="Gathering">
        <AssemblySection AssemblyIDs="Part2"/>
        <AssemblySection AssemblyIDs="Part2"/>
        <AssemblySection AssemblyIDs="Part2"/>
    </AssemblySection>
</Assembly>
<Assembly Class="Parameter" ID="A2" JobID="Book2" Order="Collecting"
    Status="Available">
    <Layout Class="Parameter" ID="L2" Status="Unavailable"/>
</ResourcePool>
<ResourceLinkPool>
    <StrippingParamsLink Usage="Input" rRef="L1"/>
    <AssemblyLink Usage="Input" rRef="A1"/>
    <AssemblyLink Usage="Input" rRef="A2"/>
    <LayoutLink Usage="Output" rRef="L2"/>
</ResourceLinkPool>
</JDF>

```

O.7 DigitalDelivery Examples

Example O-24: DigitalDelivery: Before the Delivery

Instruct the digital delivery Device to compress the files delivered in gzip compression. The part that changes has an orange tag and magenta attributes.

TBD 2.x Example.

Example O-25: DigitalDelivery: After the Delivery

Since the input **RunList** Resource is without *@Compression* and the output **RunList** Resource is with *@Compression* — it will instruct the digital delivery Device to compress the files delivered.

TBD 2.x Example.

Example O-26: Delivery and DigitalDelivery Processes

Full example of **DeliveryParams** translated to **Delivery** Processes

The following example describes:

- 1 Intent with upload file through www form and instruction to return the intermediate files in digital media together with the final product.
- 2 **Delivery** Process sub-jdf describing the upload to ftp server + compression + storage.
- 3 **Delivery** Process sub-jdf describing the return of final product and digital media via Fedex with values for service level and tracking id.

TBD 2.x Example.

```
<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1"
    DescriptiveName="ArtDeliveryIntent translated to Delivery and
    DigitalDelivery processes" ID="ID000"
    Status="InProgress" Type="Product" JobPartID="ID879" Version="1.4"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```

<ResourcePool>
  <NodeInfo ID="N01" Class="Parameter" Status="Available"
    JobPriority="100"/>
  <CustomerInfo ID="Cus01" Class="Parameter" Status="Available"
    CustomerJobName="Job title ...">
    <Contact ContactTypes="Customer">
      <Address City="Alta" PostalCode="36930" Region="AV"
        Street="123 Gibrish Street"/>
      <Person FamilyName="Spencer" FirstName="Ron"/>
      <ComChannel ChannelType="Phone" ChannelUsage="DayTime"
        Locator="tel:+44-019-1234-4567"/>
      <ComChannel ChannelType="Fax"
        ChannelUsage="Business DayTime NightTime"
        Locator="tel:+44-019-1234-4567"/>
    </Contact>
  </CustomerInfo>
  <ArtDeliveryIntent Class="Intent" ID="Link002"
    ReturnList="DigitalMedia" Status="Available">
    <ArtHandling DataType="EnumerationSpan"
      Range="Return ReturnWithProduct"/>
    <ReturnMethod DataType="NameSpan" Preferred="FedEx"/>
    <ArtDelivery ArtDeliveryType="DigitalNetwork">
      <Contact ContactTypes="Delivery">
        <ComChannel ChannelType="WWW" ChannelTypeDetails="Form"
          Locator="http://www.server.com/uploader.aspx"/>
      </Contact>
      <RunList>
        <LayoutElement>
          <FileSpec
            URL="file:///D:/WINNT/Profiles/23423/Desktop/test.pdf"/>
        </LayoutElement>
      </RunList>
    </ArtDelivery>
  </ArtDeliveryIntent>
  <Contact Class="Parameter" ContactTypes="Delivery" ID="Shipping001"
    Status="Available">
    <Address City="Alta" PostalCode="36930" Region="AV"
      Street="123 Gibrish Street"/>
    <Person FamilyName="Jones" FirstName="Bill"/>
    <ComChannel ChannelType="Phone" ChannelTypeDetails="Mobile"
      Locator="tel:+44-078-1234-4567"/>
  </Contact>
  <Component Amount="500" Class="Quantity" ComponentType="FinalProduct"
    ID="ItemFinal" Status="Unavailable"/>
</ResourcePool>
<ResourceLinkPool>
  <ArtDeliveryIntentLink Usage="Input" rRef="Link002"/>
  <ComponentLink Amount="500" Usage="Output" rRef="ItemFinal"/>
  <NodeInfoLink Usage="Input" rRef="N01"/>
  <CustomerInfoLink Usage="Input" rRef="Cus01"/>
</ResourceLinkPool>
<JDF ID="J171373" Status="Completed" Type="DigitalDelivery"
  JobPartID="ID877" >
  <ResourcePool>
    <CustomerInfo ID="Cus02" Class="Parameter" Status="Available"
      CustomerJobName="Job title ..."/>
    <RunList Class="Parameter" ID="FileListLink1" Status="Available">
      <LayoutElement>

```

```

<FileSpec
    URL="file:///D:/WINNT/Profiles/23423/Desktop/test.pdf"/>
</LayoutElement>
</RunList>
<DigitalDeliveryParams Class="Parameter"
    DigitalDeliveryDirection="Push"
    DigitalDeliveryProtocol="FTP"
    ID="DestinationLink" Method="WebServer"
    Status="Available">
    <Contact ContactTypes="Delivery">
        <ComChannel ChannelType="WWW" ChannelTypeDetails="Form"
            Locator="http://www.server.com/uploader.aspx"/>
    </Contact>
    <Contact ContactTypes="Sender">
        <ComChannel ChannelType="Email"
            Locator="mailto:sender@email.com"/>
    </Contact>
</DigitalDeliveryParams>
<RunList Class="Parameter" ID="FileListLink2" Status="Available">
    <Disposition MinDuration="P30D"/>
    <LayoutElement>
        <FileSpec Compression="Deflate" URL="test.pdf">
            <Container>
                <FileSpec MimeType="application/zip"
                    URL="file://network_share/uploaded%20files/test.zip"/>
            </Container>
        </FileSpec>
    </LayoutElement>
</RunList>
</ResourcePool>
<ResourceLinkPool>
    <DigitalDeliveryParamsLink Usage="Input" rRef="DestinationLink"/>
    <RunListLink Usage="Input" rRef="FileListLink1"/>
    <CustomerInfoLink Usage="Input" rRef="Cus02"/>
    <RunListLink Usage="Output" rRef="FileListLink2"/>
</ResourceLinkPool>
<AuditPool>
    <PhaseTime DescriptiveName="Upload of Job 171373 to Server"
        End="2003-01-08T12:27:56Z" Start="2003-01-08T12:27:40Z"
        Status="InProgress"
       TimeStamp="2003-01-08T12:27:56Z"/>
    <Created AgentName="Server uploader 1.51" TimeStamp="2003-01-08T12:27:40Z"/>
    <ProcessRun End="2003-01-08T12:27:56Z" EndStatus="Completed"
        Start="2003-01-08T12:27:40Z" TimeStamp="2003-01-08T12:27:56Z"/>
</AuditPool>
</JDF>
<JDF DescriptiveName="The Return of product and digital media with intermediate
    materials"
    ID="X00000" Status="Waiting" Type="Delivery" JobPartID="ID878" >
    <ResourceLinkPool>
        <ComponentLink Usage="Output" rRef="Item001"/>
        <DigitalMediaLink Usage="Output" rRef="Item002"/>
        <DeliveryParamsLink Usage="Input" rRef="Delivery001"/>
    </ResourceLinkPool>
    <ResourcePool>
        <RunList Class="Parameter" ID="FileListLink0" PartIDKeys="Run"
            Status="Available">
            <RunList Run="1">

```

```

<LayoutElement>
    <FileSpec URL=".//ForReturn/Intermediate/test.pdf"/>
</LayoutElement>
</RunList>
<RunList Run="2">
    <LayoutElement>
        <FileSpec URL=".//ForReturn/Final/test.pdf"/>
    </LayoutElement>
</RunList>
</RunList>
<Component Amount="500" Class="Quantity" ComponentType="FinalProduct"
    ID="Item001" ProductID="AG5678" Status="Available" Unit="1"/>
<DigitalMedia Amount="1" Capacity="700" Class="Handling" ID="Item002"
    MediaLabel="TempResults" MediaType="CD" Status="Available">
    <RunListRef rRef="FileListLink0"/>
</DigitalMedia>
<DeliveryParams Class="Parameter" ID="Delivery001" Status="Available">
    <Drop Method="FedEx" ServiceLevel="Ground" TrackingID="1234567890Z">
        <ContactRef rRef="Shipping001"/>
        <DropItem Amount="500" Unit="1">
            <ComponentRef rRef="Item001"/>
        </DropItem>
        <DropItem Amount="1">
            <DigitalMediaRef rRef="Item002"/>
        </DropItem>
    </Drop>
</DeliveryParams>
</ResourcePool>
</JDF>
</JDF>

```

Example O-27: Full Example of Digital Delivery through Central Server

The following example describes:

- 1 Upload of files to server by FTP protocol
- 2 Request for 10 days storage on server
- 3 Request to Mac Binary encode the files on server
- 4 Send to multiple destinations: 1 is email address and 1 is registered address in a private directory
- 5 Download of files by HTTP protocol
- 6 Decode from Mac Binary when downloading to target
- 7 Download by 1 destination out of the 2.

TBD 2.x Example.

```

<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1"
    DescriptiveName="Digital Delivery through central server;
        example with Process Group"
    ID="ID000" Status="InProgress" Type="ProcessGroup" JobPartID="ID200"
    Version="1.4" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<ResourcePool>
    <NodeInfo ID="Node001" Class="Parameter" Status="Available" JobPriority="60"/>
    <Contact Class="Parameter" ContactTypes="Delivery" ID="DestLink"
        PartIDKeys="Location" Status="Available">
        <Contact Location="Dest1">
            <ComChannel ChannelType="Email"

```

```

        Locator="mailto:Reciever1@hotmail.com"/>
    </Contact>
    <Contact Location="Dest2">
        <ComChannel ChannelType="PrivateDirectory"
            ChannelTypeDetails="VioAddress"
            Locator="Best Workgroup@Best Company"/>
    </Contact>
</Contact>
<RunList Class="Parameter" ID="TempFileListLink" PartIDKeys="Run"
    Status="Available">
    <Disposition MinDuration="P10D"/>
    <RunList Run="1">
        <LayoutElement>
            <FileSpec Compression="MacBinary"
                URL=".~/Atlas/Europe.bmp.bin"/>
        </LayoutElement>
    </RunList>
    <RunList Run="2">
        <LayoutElement>
            <FileSpec Compression="MacBinary"
                URL=".~/Atlas/America.jpg.bin"/>
        </LayoutElement>
    </RunList>
</RunList>
</ResourcePool>
<JDF DescriptiveName="Upload Job to Server" ID="ID001" JobID="J702555"
    Status="Completed" Type="DigitalDelivery" JobPartID="ID201">
<ResourcePool>
    <CustomerInfo ID="Cus001" Class="Parameter" Status="Available"
        CustomerJobName="World atlas maps #2"/>
    <RunList Class="Parameter" Directory="file:///c:/MyDir/JobForSend"
        ID="SourceFileListLink0" PartIDKeys="Run" Status="Available">
        <RunList Run="1">
            <LayoutElement>
                <FileSpec FileSize="240066" URL=".~/Atlas/Europe.bmp"/>
            </LayoutElement>
        </RunList>
        <RunList Run="2">
            <LayoutElement>
                <FileSpec FileSize="33947" URL=".~/Atlas/America.jpg"/>
            </LayoutElement>
        </RunList>
    </RunList>
    <Contact Class="Parameter" ContactTypes="Sender" ID="SendLink"
        Status="Available">
        <ComChannel ChannelType="Email"
            Locator="mailto:sender@email.com"/>
    </Contact>
    <DigitalDeliveryParams Class="Parameter"
        DigitalDeliveryDirection="Push"
        DigitalDeliveryProtocol="FTP"
        ID="DestinationLink0" Method="Vio"
        PartIDKeys="Location" Status="Available">
        <Comment Name="Instruction">
            Please take these maps and add them to the rest ...
        </Comment>
        <DigitalDeliveryParams Location="SenderToDest1">
            <ContactRef rRef="SendLink"/>

```

```

<ContactRef rRef="DestLink">
    <Part Location="Dest1"/>
</ContactRef>
</DigitalDeliveryParams>
<DigitalDeliveryParams Location="SenderToDest2">
    <ContactRef rRef="SendLink"/>
    <ContactRef rRef="DestLink">
        <Part Location="Dest2"/>
    </ContactRef>
</DigitalDeliveryParams>
</DigitalDeliveryParams>
</ResourcePool>
<ResourceLinkPool>
    <DigitalDeliveryParamsLink Usage="Input" rRef="DestinationLink0"/>
    <CustomerInfoLink Usage="Input" rRef="Cus001"/>
    <NodeInfoLink Usage="Input" rRef="Node001"/>
    <RunListLink Usage="Input" rRef="SourceFileListLink0"/>
    <RunListLink Usage="Output" rRef="TempFileListLink"/>
</ResourceLinkPool>
<AuditPool>
    <ProcessRun DescriptiveName="Upload of Job 702555 to Vio Server"
        End="2002-07-21T10:47:11Z" EndStatus="Completed"
        Start="2002-07-21T10:45:52Z"
        TimeStamp="2002-07-21T10:47:11Z"/>
    <Created AgentName="Vio Server 4.3" TimeStamp="2002-07-21T10:45:52Z"/>
</AuditPool>
</JDF>
<JDF DescriptiveName="Download Job from Server to destination" ID="ID002"
    JobID="J702555" Status="Pool" Type="DigitalDelivery"
    JobPartID="ID202">
<ResourcePool>
    <RunList Class="Parameter" Directory="File:///e:/My%20Download"
        ID="TargetFileListLink1" PartIDKeys="Run" Status="Available">
        <RunList Run="1">
            <LayoutElement>
                <FileSpec FileSize="240066" URL=".//Atlas/Europe.bmp"/>
            </LayoutElement>
        </RunList>
        <RunList Run="2">
            <LayoutElement>
                <FileSpec FileSize="33947" URL=".//Atlas/America.jpg"/>
            </LayoutElement>
        </RunList>
    </RunList>
    <DigitalDeliveryParams Class="Parameter"
        DigitalDeliveryDirection="Pull"
        DigitalDeliveryProtocol="HTTP"
        ID="DestinationLink1" Method="Vio"
        PartIDKeys="Location" Status="Available">
        <DigitalDeliveryParams Location="ToDest1">
            <ContactRef rRef="DestLink">
                <Part Location="Dest1"/>
            </ContactRef>
        </DigitalDeliveryParams>
        <DigitalDeliveryParams Location="ToDest2">
            <ContactRef rRef="DestLink">
                <Part Location="Dest2"/>
            </ContactRef>

```

```

        </DigitalDeliveryParams>
    </DigitalDeliveryParams>
</ResourcePool>
<ResourceLinkPool>
    <DigitalDeliveryParamsLink Usage="Input" rRef="DestinationLink1"/>
    <RunListLink Usage="Input" rRef="TempFileListLink"/>
    <NodeInfoLink Usage="Input" rRef="Node001"/>
    <RunListLink Usage="Output" rRef="TargetFileListLink1"/>
</ResourceLinkPool>
<StatusPool Status="InProgress">
    <PartStatus Status="Completed">
        <Part Location="ToDest2"/>
    </PartStatus>
</StatusPool>
<AuditPool>
    <Created AgentName="Vio Server 4.3"TimeStamp="2002-07-21T10:48:57Z"/>
    <ProcessRun DescriptiveName="HTTP Download of Job by
        Best Workgroup@Best Company"
        End="2002-07-21T10:50:11Z" EndStatus="Completed"
        Start="2002-07-21T10:48:57Z"
        TimeStamp="2002-07-21T10:50:11Z">
        <Part Location="ToDest2"/>
    </ProcessRun>
</AuditPool>
</JDF>
</JDF>

```

O.8 Automated Imposition

Example O-28: Algorithm for Processing an Imposition Template

The pseudocode below describes how a Page Pool or Page Pool List might be processed through an Imposition Template. Note that the algorithm described is a fairly lazy algorithm that relies on detecting that no content can be placed on a sheet to detect end of content. In addition, cut and stack and `@BaseOrdReset = "PagePoolList"` are not supported in this example:

Note: numPagesInPagePool and lastPagePoolPositiveIndex will be affected by execution of `Layout/` `PageCondition` Elements.

TBD 2.x Example.

Example O-29: Format of Variable Data Structured Content

The BNF and example below describe a generic variable data structured content format using XML to represent the structure elements. This format is able to represent the most common attributes of existing variable data languages, and will be used to describe the input data sets for this section's examples:

The following is BNF for eVDPM (example Variable Document Print Markup Language) XML:

Page ::= [Metadata*] - represents the graphical content of a single page which may contain metadata.

(**Note:** Metadata was added to the Page Element in the meeting discussion)

Metadata - represents arbitrary key/value metadata information

DocPart ::= [Metadata*] [DocPart+ || Page+]

(**Note:** cannot have both DocPart and Page elements in the same DocPart element.)

Record ::= [Metadata*] [DocPart+ || Page+]

(**Note:** Both DocPart and Page elements SHALL not be specified in the same Record.)

RecordGroup ::= [Metadata] [RecordGroup+ || Record+]

(**Note:** all Record elements SHALL be specified at the same RecordGroup node hierarchical level)

Below is an example structure using eVDPMML syntax (which is NOT XJDF syntax):

TBD 2.x Example.

Example O-30: Page Pools

All recipients receive a one page cover letter printed on substrate A and a personalized document containing two or more customized two sided pages printed on substrate B and all pages to be corner stapled together.

Demonstrates use of Partitioning for mapping Page Pools of two different documents to independent sets of sheets relying on the structure of the PDL.

The eVDPMML is:

TBD 2.x Example.

```
<RecordGroup>
  <Record>
    <Metadata Key="RecID" Value="0"/>
    <DocPart>
      <Metadata Key="Part" Value="CoverLetter"/>
      <Page/>
    </DocPart>
    <DocPart>
      <Metadata Key="Part" Value="Brochure"/>
      <Page/>
      <Page/>
      <Page/>
      <Page/>
    </DocPart>
  </Record>
  <Record>
    <Metadata Key="RecID" Value="1"/>
    <DocPart>
      <Metadata Key="Part" Value="CoverLetter"/>
      <Page/>
    </DocPart>
    <DocPart>
      <Metadata Key="Part" Value="Brochure"/>
      <Page/>
      <Page/>
      <Page/>
      <Page/>
    </DocPart>
  </Record>
  <!-- ... -->
  <Record>
    <Metadata Key="RecID" Value="n"/>
    <DocPart>
      <Metadata Key="Part" Value="CoverLetter"/>
      <Page/>
    </DocPart>
    <DocPart>
      <Metadata Key="Part" Value="Brochure"/>
      <Page/>
    </DocPart>
```

```

<Page/>
</DocPart>
</Record>
</RecordGroup>
```

The XJDF is:

TBD 2.x Example.

```

<RunList Class="Parameter" ID="Ex3_RunList" Status="Available">
    <LayoutElement>
        <FileSpec URL="MyVDPRecords.vdpml"
            MimeType="application/x-evdpml+xml"/>
    </LayoutElement>
    <MetadataMap DataType="PartIDKeys" Name="RunTags"
        ValueFormat="%s" ValueTemplate="doctype" Context="Document">
        <Expr Name="doctype" Value="CoverLetter">
            <and>
                <NameEvaluation
                    Path="Metadata/@Key" RegExp="Part"/>
                <NameEvaluation
                    Path="Metadata/@Value" RegExp="CoverLetter"/>
            </and>
        </Expr>
        <Expr Name="doctype" Value="Brochure">
            <and>
                <NameEvaluation
                    Path="Metadata/@Key" RegExp="Part"/>
                <NameEvaluation
                    Path="Metadata/@Value" RegExp="Brochure"/>
            </and>
        </Expr>
    </MetadataMap>
</RunList>
<Media Class="Consumable" ID="Medial" Status="Available"/>
<!--
    NOTE: MetadataMap(s) are applied to each document
    node input to the Imposition process independent of each other.
-->
<Layout Class="Parameter" ID="Ex2_Layout" Status="Available"
    PartIDKeys="RunTags SheetName Side" Automated="true"
    LockOrigins="true" BaseOrdReset="PagePool">
    <MediaRef rRef="Medial"/>
<!-- Used by Imposition process to obtain media dimensions -->
<Layout RunTags="CoverLetter">
    <Layout SheetName="LetterSheet">
        <Layout Side="Front">
            <ContentObject CTM="1 0 0 1 0 0" Ord="0"/>
            <!-- First page of CoverLetter DocPart -->
        </Layout>
    </Layout>
<Layout RunTags="Brochure">
    <Layout SheetName="BrochureSheets">
        <Layout Side="Front">
            <ContentObject CTM="1 0 0 1 0 0" Ord="0"/>
            <!--Front side of Brochure sheet -->
        </Layout>
        <Layout Side="Back">
            <ContentObject CTM="1 0 0 1 0 0" Ord="1"/>
```

```
<!--Back side of Brochure sheet -->
</Layout>
</Layout>
</Layout>
</Layout>
```

Example O-31: Booklet Using Automated Imposition

All recipients receive a single customized saddle stitched booklet where the two-page cover (front and back outside only) and the four or more body pages are also printed on substrate A using cut and stack production. The finished sheet size of each component of a booklet is 8.5" x 11" and the finished booklet is 5.5" x 8.5". The production of the component will be performed cut and stack on 17" x 11". Two saddle stitch imposed page pairs will be generated per sheet definition.

The eVDPML is:

TBD 2.x Example.

The XJDF is:

TBD 2.x Example.

Appendix P Deprecated Elements, JMF Messages, Processes and Resources

Nothing has been deprecated.

```
""""<Query ID="M170" Type="Events" xsi:type="QueryEvents">
    <Subscription URL="http://www.anycompany.com/MIS/JMF/JobTracker"/>
    <NotificationFilter Classes="Event Warning Error Fatal" SignalTypes="All"/>
</Query>
<Response ID="M1001" refID="M170" Type="Events" xsi:type="ResponseEvents"
    xmlns:anycompany="http://www.anycompany.com">
    <NotificationDef Classes="Warning Error Fatal" Type="Error"/>
    <NotificationDef Classes="Event" Type="FCNKey"/>
    <NotificationDef Classes="Event Error" Type="Barcode"/>
    <NotificationDef Classes="Event" Type="SystemTimeSet"/>
    <NotificationDef Classes="Event" Type="anycompany:PrivateEvent_1"/>
    <NotificationDef Classes="Event" Type="anycompany:PrivateEvent_2"/>
    <NotificationDef Classes="Event" Type="anycompany:PrivateEvent_2"/>
    <NotificationDef SignalType="Status"/>
    <NotificationDef SignalType="Resource"/>
</Response>
<Query ID="Q1" Type="KnownControllers" SenderID="MIS"
    xsi:type="QueryKnownControllers">
    <ControllerFilter ControllerID="PrintController1" URLTypes="JMF SecureJMF"/>
</Query>
<Response ID="M1" Type="KnownControllers" xsi:type="ResponseKnownControllers"
    refID="Q1" SenderID="RegistrationServer">
    <JDFController ControllerID="PrintController1"
        DescriptiveName="Printer Controller"
        URL="http://www.anycompany.com/controller"
        URLType="JMF"/>
    <JDFController ControllerID="PrintController1"
        DescriptiveName="Printer Controller"
        URL="https://www.anycompany.com/controller/secure"
        URLType="SecureJMF"/>
</Response>
<Response ID="RepMsg" Type="RepeatMessages" xsi:type="ResponseRepeatMessages">
    <Response ID="R1" Time="2000-06-14T11:00:01+02:00" Type="Status"
        xsi:type="ResponseStatus"/>
    <Response ID="R2" Time="2000-06-14T10:50:22+02:00" Type="Occupation"
        xsi:type="ResponseOccupation"/>
    <Signal ID="R3" Time="2000-06-14T08:20:23+02:00" Type="Resource"
        xsi:type="SignalResource"/>
    <Signal ID="R4" Time="2000-06-14T03:01:22+02:00" Type="Notification"
        xsi:type="SignalNotification"/>
</Response>
<Response ID="M1" Type="Occupation" xsi:type="ResponseOccupation" refID="Q1">
    <!--Two Jobs on one Device with one operator-->
    <Occupation Busy="30" JobID="J1">
        <Employee PersonalID="P1234"/>
        <Device DeviceID="Press1"/>
    </Occupation>
    <Occupation Busy="70" JobID="J2">
        <Employee PersonalID="P1234"/>
        <Device DeviceID="Press1"/>
    </Occupation>
    <!--Another operator on Job j2 -->
    <Occupation Busy="50" JobID="J2">
        <Employee PersonalID="P4321"/>

```

```
<Device DeviceID="Press1"/>
</Occupation>
<!--No Job context -->
<Occupation Busy="0">
    <Device DeviceID="Press2"/>
    <Employee PersonalID="P5678"/>
</Occupation>
</Response>
<Response ID="M1" Type="Track" xsi:type="ResponseTrack" refID="Q1">
    <TrackResult IsDevice="true" JobID="1" JobPartID="42"
        URL="http://www.anycompany.com/controller"/>
</Response>
```

Appendix Q List of Figures

Figure 1-1 Handling of Default Values of JDF Attributes	18
Figure 2-1 Example of JDF and JMF workflow interactions	29
Figure 2-2 Example of XJDF and XJMF workflow interactions	30
Figure 2-3 JDF tree structure	32
Figure 2-4 Example of a hierarchical tree structure of JDF Nodes	34
Figure 2-5 Example of a Process chain linked by Input Resources and Output Resources	34
Figure 2-6 Standard coordinate system	36
Figure 2-7 Relation between Resource and process coordinate systems	36
Figure 2-8 Relation between Resource and process coordinate systems	37
Figure 2-9 Layout of simple saddle stitched brochure (product example)	41
Figure 2-10 Equation for Surface Coordinate System Transformations	41
Figure 2-11 Surface coordinate system	41
Figure 2-12 Press coordinate system used for Sheet-Fed Printing	42
Figure 2-13 Press coordinate system used for Web Printing	42
Figure 2-14 Coordinate systems after Folding (product example)	43
Figure 2-15 Coordinate systems after Collecting (product example)	43
Figure 2-16 Examples of Transformations and Coordinate Systems in JDF XJDF.	45
Figure 2-17 Transforming a point (example)	47
Figure 3-1 Any-Element (generic content) – a diagram of its structure	52
Figure 3-2 JDF Node – a Diagram of its Structure	62
Figure 3-3 XJDF Node – a Diagram of its Structure	63
Figure 3-4 Job hierarchy with Process, Process Group and Product Intent Nodes	68
Figure 3-5 Combined Process Node dependencies	74
Figure 3-6 ResourcePool and Abstract Resource Element – a diagram of the structure	83
Figure 3-7 ResourceSet – a Diagram of its Structure	86
Figure 3-8 AmountPool – a Diagram of its Structure	95
Figure 3-9 Nodes linked by a Resource	102
Figure 3-10 ResourceLink Elements and ResourceRef Elements	103
Figure 3-11 ResourceLinkPool and ResourceLink Element – a diagram of the structure	105
Figure 3-12 AmountPool – a Diagram of its Structure	112
Figure 3-13 Amount handling	123
Figure 3-14 Workflow for splitting shared Input Resources	150
Figure 3-15 Workflow for combining shared Output Resources	150
Figure 3-16 Workflow for splitting independent Input Resources	151
Figure 3-17 Workflow for combining independent Output Resources	151
Figure 3-18 AuditPool and Abstract Audit Element – a diagram of the structure	153
Figure 3-19 Specific Audit – a Diagram of its Structure	157
Figure 4-1 Life Cycle of a JDF Node	187
Figure 4-2 Example of a simple Process chain linked by Resources	188
Figure 4-3 Example of a simple Process chain linked by Resources	189
Figure 4-4 Example of a Pipe Resource Linking Two Processes via Pull	192
Figure 4-5 Example of a Pipe Resource Linking Two Processes via Push	192
Figure 4-6 Example of status transitions in case of overlapping Processing	193
Figure 4-7 Example of status transitions in case of overlapping Processing	193
Figure 4-8 The spawning and merging mechanism and its phases	199
Figure 4-9 JDF Node structure that requires Resource copying during spawning and merging	201

Figure 4-10 Example for a JDF Node structure with nested spawning	203
Figure 4-11 Parameter space in Device capabilities	206
Figure 5-1 JMF Root Element – a diagram of its structure	212
Figure 5-2 Interaction of Messages with a Subscription	217
Figure 5-3 Interaction of Command and Acknowledge Messages	227
Figure 5-4 Example of Reliable Signaling	230
Figure 5-5 Without UpdateJDF Message	299
Figure 5-6 With UpdateJDF Message	299
Figure 5-7 Mechanism of a PipePull Message	308
Figure 5-8 Mechanism of a PipePush Message	309
Figure 5-9 JMF QueueEntry Status Transition Diagram	314
Figure 5-10 XJMF QueueEntry Status Transition Diagram	316
Figure 5-11 Effects of the global queue Messages on the queue Status	337
Figure 6-1 Imposition for Cut and Stack	383
Figure 6-2 Worst case scenario for area coverage calculation	391
Figure 6-3 Overview of Web Printing	401
Figure 6-4 Bundle Creation	408
Figure 6-5 Bundle Transport	408
Figure 6-6 Combined Process with Feeding Process	415
Figure 6-7 Input Components	415
Figure 6-8 Output Component	415
Figure 6-9 Gathering	417
Figure 6-10 Print Roll	424
Figure 6-11 Packaging Process Coordinate System	435
Figure 7-1 ProductList– a Diagram of its Structure	442
Figure 7-2 Roll-up Display	457
Figure 7-3 Structure of a normal hardcover book	466
Figure 7-4 Structure of a padded hardcover book	466
Figure 7-5 Structure of a book with GlueProcedure = "SideOnly" (Layflat)	472
Figure 8-1 BinderySignature Trims	535
Figure 8-2 WebCellAlignment, Example 1	536
Figure 8-3 WebCellAlignment Example 2	537
Figure 8-4 WebCellAlignment Example 3	538
Figure 8-5 Definition of margins in SignatureCell	543
Figure 8-6 Tightbacking for Block Preparation	543
Figure 8-7 Rounding and Backing for Block Preparation	544
Figure 8-8 Folding examples for some values of BoxFoldAction/@Action	547
Figure 8-9 BoxFoldingType Attribute for values of Type00, Type01 and Type02	548
Figure 8-10 BoxFoldingType Attribute for values of Type03, Type04 and Type10	548
Figure 8-11 BoxFoldingType Attribute for values of Type 11, Type12 and Type13	549
Figure 8-12 BoxFoldingType Attribute for values of Type15 and Type20	549
Figure 8-13 Box packing	551
Figure 8-14 BundlingParams Coordinate System	555
Figure 8-15 CaseMakingParams	558
Figure 8-16 Parameters and coordinate system for CasingIn	559
Figure 8-17 Parameters used for channel binding	560
Figure 8-18 Coordinate systems used for collecting	563
Figure 8-19 Component – terms and definitions	588
Figure 8-20 Orientation of the Finished Product on the Roll	593
Figure 8-21 Parameters and coordinate system for cover application	612

Figure 8-22 Parameters and coordinate system for glue application	613
Figure 8-23 Definition of the PlatePosition Attribute on a newspaper-Web Press	621
Figure 8-24 Example of a single physical section of eight pages	621
Figure 8-25 Basic Shape for RepeatDesc/@LayoutStyle Examples	640
Figure 8-26 RepeatDesc/@LayoutStyle = "StraightNest"	640
Figure 8-27 RepeatDesc/@LayoutStyle = "Reverse2ndRow"	641
Figure 8-28 RepeatDesc/@LayoutStyle = "Reverse2ndRowAligned"	641
Figure 8-29 RepeatDesc/@LayoutStyle = "Reverse2ndColumn"	642
Figure 8-30 RepeatDesc/@LayoutStyle = "Reverse2ndColumnAligned"	642
Figure 8-31 RepeatDesc/@GutterX2 and @GutterY2: Secondary Gutters	643
Figure 8-32 Parameters and coordinate system used for end-Sheet gluing	656
Figure 8-33 Names of the reference edges of a Sheet in the FoldingParams Resource	671
Figure 8-34 Fold catalog part 1	674
Figure 8-35 Fold catalog part 2	675
Figure 8-36 Coordinate system used for Gathering	678
Figure 8-37 Parameters and coordinate system for glue application	679
Figure 8-38 Parameters and coordinate system used for Inserting	708
Figure 8-39 Setup of the Jacketing Machinery	717
Figure 8-40 Parameters and coordinate system for jacketing	717
Figure 8-41 Anchor with No Scaling and No Rotation	730
Figure 8-42 Anchor with No Scaling and Rotation of 90° Clockwise	731
Figure 8-43 Anchor with 1.5 Scaling and Rotation of 90° Clockwise	732
Figure 8-44 Diagram of a 4-up cross-folded saddle-stitch imposition with vertical gutter creep	790
Figure 8-45 Diagram of a step-and-repeat 2-up saddle-stitch imposition with vertical spine gutter creep	791
Figure 8-46 Parameters used for channel binding	795
Figure 8-47 Paper Roll with some Roll-specific Information	814
Figure 8-48 Relief and Floor Thickness for a Flexo Plate or Flexo Sleeve	814
Figure 8-49 Types of Interlocks for Flexo Sleeve	814
Figure 8-50 TabSetCollationOrder Attribute Values	816
Figure 8-51 Diagram of a Single Bank of Tabs	817
Figure 8-52 Inside Loss, Outside Gain	818
Figure 8-53 PRGroup – a diagram of its structure	857
Figure 8-54 PrintRollingParams Coordinate System	868
Figure 8-55 ShapeTemplate Example 1	912
Figure 8-56 ShapeTemplate Example 2	913
Figure 8-57 ShapeTemplate Example 3	913
Figure 8-58 Parameters and coordinate systems for the SpinePreparation Process	920
Figure 8-59 Parameters and coordinate system for the SpineTaping Process	921
Figure 8-60 Stacking Layers	922
Figure 8-61 Pile Patterns	922
Figure 8-62 Odd count handling for a Bundle	923
Figure 8-63 Odd count handling for a Layer	923
Figure 8-64 Staple shapes	926
Figure 8-65 Parameters and coordinate system used for saddle stitching	926
Figure 8-66 Parameters and coordinate system used for Stitching	927
Figure 8-67 Stitching Coordinate System for StitchOrigin Values	927
Figure 8-68 Strapped Bundle	931
Figure 8-69 Strapped Bundle with Sub-bundles	931

Figure 8-70 Shingling for Stripping	936
Figure 8-71 Shingling for Stripping – Details	936
Figure 8-72 RelativeBox including margins	937
Figure 8-73 Definition of margins in StripCellParams	940
Figure 8-74 Parameters and coordinate system used for thread sewing	947
Figure 8-75 Parameters and coordinate system used for side sewing	947
Figure 8-76 Parameters and coordinate system used for trimming	959
Figure 9-1 DeviceCap – a diagram of its structure	970
Figure 9-2 DeviceCap – a Diagram of its Structure	974
Figure 9-3 Abstract State Element – a diagram of its structure	982
Figure 9-4 Specific State – a Diagram of its Structure	987
Figure 9-5 macro Element – a diagram of its structure	1016
Figure 9-6 Abstract Term Element – a diagram of its structure	1020
Figure 9-7 Term– a Diagram of its Structure	1021
Figure 9-8 Abstract Evaluation Element – a diagram of its structure	1024
Figure 9-9 Specific Evaluation – a Diagram of its Structure	1027
Figure 10-1 Anchor with No Scaling and No Rotation	1086
Figure 10-2 Anchor with No Scaling and Rotation of 90° Clockwise	1086
Figure 10-3 Anchor with 1.5 Scaling and Rotation of 90° Clockwise	1087
Figure 10-4 Hole line parameters	1096
Figure 10-5 Line hole punching for multiple webs	1096
Figure 10-6 RegisterRibbon lengths and coordinate system for BlockPreparation	1114
Figure 11-1 Life Cycle of a Process	1120
Figure 11-2 Example of a Pipe Resource Linking Two Processes via Pull	1122
Figure 11-3 Example of a Pipe Resource Linking Two Processes via Push	1122
Figure 11-4 Example of status transitions in case of overlapping Processing	1123
Figure 11-5 Example of status transitions in case of overlapping Processing	1123
Figure 11-6 Example of Exchange of Certificates	1130
Figure 11-7 Interaction of Messages with a Subscription	1132
Figure 11-8 Example of Reliable Signaling	1133
Figure J-1 Legend for Interpreting Diagrams	1207
Figure P-1 Example of the spawning and merging of independent Jobs	1340
Figure P-2 Parameters and coordinate system for glue application	1371
Figure P-3 Staple shapes	1395
Figure P-4 Parameters and coordinate system used for side sewing	1396

Appendix R List of Tables

Table 0-1 Callout Icon Usage.....	xlv
Table 1-1 Basic References	4
Table 1-2 Modification Notes	6
Table 1-3 Cardinality Symbols	7
Table 1-4 Template for Element Descriptions	8
Table 1-5 Glossary	9
Table 1-6 Conformance Terminology	16
Table 1-7 JDF XJDF Data Types	20
Table 1-8 Units Used in JDF XJDF	23
Table 2-1 Information contained in JDF Nodes, arranged numerically	32
Table 2-2 Information contained in JDF Nodes, arranged by group	33
Table 2-3 Data types for specifying coordinates and transformation	37
Table 2-4 Matrices and Orientation values for describing the orientation of a Component	39
Table 2-5 JDF XJDF Processes used for the production of the simple brochure	40
Table 3-1 Any Element (generic content)	50
Table 3-2 Definition of “XXX”	54
Table 3-3 Behavior for Activation Values inTable 3-4	54
Table 3-4 JDF XJDF Node	55
Table 3-5 Comment Element	63
Table 3-6 GeneralID Element.....	66
Table 3-7 AncestorPool Element.....	75
Table 3-8 Ancestor Element.....	75
Table 3-9 ResourcePool Element	77
Table 3-10 Abstract Resource Element.....	78
Table 3-11 SourceResource Element	82
Table 3-12 ProductList Element	84
Table 3-13 ResourceSet Element	84
Table 3-14 Dependent Element	87
Table 3-15 Part Element	88
Table 3-16 AmountPool Element	95
Table 3-17 PartAmount Element	96
Table 3-18 WasteDetails Attribute Values	96
Table 3-19 Abstract Parameter Resource Element.....	98
Table 3-20 Abstract PhysicalResource Element	99
Table 3-21 Location Element	100
Table 3-22 ResourceLinkPool Element.....	102
Table 3-23 ResourceLink Element.....	105
Table 3-24 ProcessUsage Attribute Values	110
Table 3-25 AmountPool Element.....	112
Table 3-26 PartAmount Element.....	113
Table 3-27 Lot Element	117
Table 3-28 Abstract ResourceElement.....	118
Table 3-29 Abstract ResourceRef Element.....	119
Table 3-30 Example of actual amount and amount handling	123
Table 3-31 Identical Element.....	130
Table 3-32 Partitionable Resource Element.....	133
Table 3-33 Part Element	134

Table 3-34 Condition Attribute Values	144
Table 3-35 PartUsage Attribute examples.....	147
Table 3-36 AuditPool Element	152
Table 3-37 Abstract Audit Element	154
Table 3-38 List of Audit Elements	154
Table 3-39 Created Audit Element	158
Table 3-40 Deleted Audit Element.....	158
Table 3-41 Merged Audit Element.....	159
Table 3-42 Modified Audit Element	159
Table 3-43 AuditNotification Element	160
Table 3-44 Notification Audit Element	160
Table 3-45 PhaseTime Audit Element	163
Table 3-46 AuditStatus Element	165
Table 3-47 Activity Element	166
Table 3-48 ModulePhase Element	166
Table 3-49 ProcessRun Audit Element	168
Table 3-50 AuditResource Element	170
Table 3-51 ResourceAudit Audit Element	170
Table 3-52 Spawed Audit Element.....	174
Table 3-53 Excerpt from TrappingParams	177
Table 4-1 Examples of Resource and Process states in the case of simple Process routing .	189
Table 4-2 Examples of Partitioning across multiple Resources	190
Table 4-3 Actions generated when a dynamic-pipe buffer passes various levels	195
Table 4-4 Event Sequence in Digital Finishing	196
Table 5-1 JMF XJMF Element	208
Table 5-2 Message Element	210
Table 5-3 List of JMF Messages	213
Table 5-4 List of XJMF Messages.....	215
Table 5-5 Query Message Element	217
Table 5-6 Subscription Element	218
Table 5-7 Command Message Element Family	220
Table 5-8 Signal Message Element Family	222
Table 5-9 Trigger Element.....	223
Table 5-10 ChangedPath Element.....	224
Table 5-11 Response Message Element Family	225
Table 5-12 Acknowledge Message Element	227
Table 5-13 Registration Message Element	229
Table 5-14 Subscription Element	231
Table 5-15 ObservationTarget Element	232
Table 5-16 Template for Message tables	235
Table 5-17 Messages for events and capabilities	236
Table 5-18 Messages for events and capabilities	236
Table 5-19 KnownDevices Message	237
Table 5-20 QueryKnownDevices Message	237
Table 5-21 DeviceFilter Element	238
Table 5-22 ResponseKnownDevices Message	239
Table 5-23 DeviceList Element	239
Table 5-24 SignalKnownDevices Message	240
Table 5-25 KnownMessages Message	240
Table 5-26 QueryKnownMessages Message	240

Table 5-27 KnownMsgQuParams Element	241
Table 5-28 ResponseKnownMessages Message	241
Table 5-29 MessageService Element	242
Table 5-30 SignalKnownMessages Message	244
Table 5-31 KnownSubscriptions Message	245
Table 5-32 QueryKnownSubscriptions Message	245
Table 5-33 SubscriptionFilter Element	245
Table 5-34 ResponseKnownSubscriptions Message	246
Table 5-35 SubscriptionInfo Element	247
Table 5-36 SignalKnownSubscriptions Message	247
Table 5-37 Notification Signal	248
Table 5-38 QueryNotification Message	248
Table 5-39 NotificationFilter Element	249
Table 5-40 ResponseNotification Message	250
Table 5-41 SignalNotification Message	250
Table 5-42 Notification Element	251
Table 5-43 Error Element	254
Table 5-44 Event Element	254
Table 5-45 Milestone Element	254
Table 5-46 RequestForAuthentication Command Message	255
Table 5-47 AuthenticationCmdParams Element	256
Table 5-48 Certificate Element.....	257
Table 5-49 AuthenticationResp Element.....	257
Table 5-50 RequestForAuthentication Query Message	257
Table 5-51 AuthenticationQuParams Element	258
Table 5-52 StopPersistentChannel Message	259
Table 5-53 CommandStopPersistentChannel Message	259
Table 5-54 StopPersChParams Element	260
Table 5-55 ResponseStopPersistentChannel Message	260
Table 5-56 Messages to query/affect a Job, Device or Controller	261
Table 5-57 Messages Relating to Jobs and Devices	262
Table 5-58 FlushResources Command.....	262
Table 5-59 CommandFlushResources Message	263
Table 5-60 FlushResources Query	263
Table 5-61 QueryFlushResources Message	263
Table 5-62 FlushResourceParams Element	264
Table 5-63 ResponseFlushResources Message	264
Table 5-64 FlushedResources Element	265
Table 5-65 SignalFlushResources Message	265
Table 5-66 ModifyNode Command	265
Table 5-67 ModifyNode Signal.....	265
Table 5-68 ModifyNodeCmdParams Element.....	266
Table 5-69 NewComment Element	266
Table 5-70 NewJDF Query Message	267
Table 5-71 NewJDFQuParams Element	267
Table 5-72 NewJDF Command Message	267
Table 5-73 NewJDFCmdParams Element	268
Table 5-74 IDInfo Element	268
Table 5-75 Resource Query Message	269
Table 5-76 QueryResource Message	270

Table 5-77 ResourceQuParams Element	270
Table 5-78 Resource Command Message	275
Table 5-79 CommandResource Message	275
Table 5-80 ResourceCmdParams Element	276
Table 5-81 ResponseResource Message	279
Table 5-82 ResourceInfo Element	279
Table 5-83 SignalResource Message	284
Table 5-84 ResourcePull Message	285
Table 5-85 ResourcePullParams Element	285
Table 5-86 ShutDown Message	286
Table 5-87 CommandShutDown Message	287
Table 5-88 ShutDownCmdParams Element	287
Table 5-89 ResponseShutDown Message	288
Table 5-90 Status Message	288
Table 5-91 QueryStatus Message	288
Table 5-92 StatusQuParams Element	290
Table 5-93 ResponseStatus Message	291
Table 5-94 DeviceInfo Element	291
Table 5-95 Activity Element	294
Table 5-96 JobPhase Element	295
Table 5-97 ModuleStatus Element	297
Table 5-98 SignalStatus Message	298
Table 5-99 UpdateJDF Command	300
Table 5-100 UpdateJDF Signal	300
Table 5-101 UpdateJDFCmdParams Element	300
Table 5-102 CreateLink Element	301
Table 5-103 CreateResource Element	301
Table 5-104 MoveResource Element	301
Table 5-105 RemoveLink Element	301
Table 5-106 WakeUp Message	302
Table 5-107 CommandWakeUp Message	303
Table 5-108 WakeUpCmdParams Element	303
Table 5-109 ResponseWakeUp Message	303
Table 5-110 Messages for Control of Dynamic Pipes	304
Table 5-111 Messages for Control of Dynamic Pipes	304
Table 5-112 CommandPipeControl Message	305
Table 5-113 PipeParams Element	305
Table 5-114 PipeClose Message	307
Table 5-115 PipePull Message	307
Table 5-116 PipePush Message	308
Table 5-117 PipePause Message	309
Table 5-118 ResponsePipeControl Message	310
Table 5-119 Messages for queue entry handling	311
Table 5-120 Messages for queue entry handling	311
Table 5-121 Status Transitions for QueueEntry Handling Messages	312
Table 5-122 AbortQueueEntry Message	317
Table 5-123 AbortQueueEntryParams Element	317
Table 5-124 HoldQueueEntry Message	318
Table 5-125 HoldQueueEntryParams Element	318
Table 5-126 CommandModifyQueueEntry Message	319

Table 5-127 ModifyQueueEntryParams Element	319
Table 5-128 Operation Attribute Values	320
Table 5-129 ResponseModifyQueueEntry Message	321
Table 5-130 RemoveQueueEntry Message	321
Table 5-131 RemoveQueueEntryParams Element	322
Table 5-132 RequestQueueEntry Message	322
Table 5-133 CommandRequestQueueEntry Message	322
Table 5-134 RequestQueueEntryParams Element	323
Table 5-135 ResponseRequestQueueEntry Message	323
Table 5-136 ResubmitQueueEntry Message	324
Table 5-137 CommandResubmitQueueEntry Message	324
Table 5-138 ResubmissionParams Element	325
Table 5-139 ResponseResubmitQueueEntry Message	326
Table 5-140 ResumeQueueEntry Message	327
Table 5-141 ResumeQueueEntryParams Element	327
Table 5-142 ReturnQueueEntry Message	327
Table 5-143 CommandReturnQueueEntry Message	328
Table 5-144 ReturnQueueEntryParams Element	328
Table 5-145 ResponseReturnQueueEntry Message	329
Table 5-146 SetQueueEntryPosition Message	329
Table 5-147 QueueEntryPosParams Element	329
Table 5-148 SetQueueEntryPriority Message	330
Table 5-149 QueueEntryPriParams Element	330
Table 5-150 SubmitQueueEntry Message	331
Table 5-151 CommandSubmitQueueEntry Message	331
Table 5-152 QueueSubmissionParams Element	332
Table 5-153 ResponseSubmitQueueEntry Message	334
Table 5-154 SuspendQueueEntry Message	335
Table 5-155 SuspendQueueEntryParams Element	335
Table 5-156 Messages for global handling of queues	336
Table 5-157 Messages for global handling of queues	336
Table 5-158 Definition of the Queue Status Attribute Values	336
Table 5-159 CloseQueue Message	337
Table 5-160 FlushQueue Command Message	338
Table 5-161 FlushQueueParams Element	338
Table 5-162 FlushQueue Query Message	338
Table 5-163 FlushQueueInfo Element	339
Table 5-164 HoldQueue Message	339
Table 5-165 OpenQueue Message	339
Table 5-166 QueueStatus Message	340
Table 5-167 QueryQueueStatus Message	340
Table 5-168 QueueStatusParams Element	340
Table 5-169 ResponseQueueStatus Message	341
Table 5-170 SignalQueueStatus Message	341
Table 5-171 ResumeQueue Message	342
Table 5-172 SubmissionMethods Message	342
Table 5-173 SubmissionMethods Element	342
Table 5-174 Queue Element	343
Table 5-175 QueueEntry Element	345
Table 5-176 QueueEntryDef Element	347

Table 5-177 QueueFilter Element	348
Table 5-178 Messages for Gang Jobs	350
Table 5-179 Messages for Gang Jobs	350
Table 5-180 Contents of the ForceGang Command Message	350
Table 5-181 CommandForceGang Message	350
Table 5-182 GangCmdFilter Element	351
Table 5-183 ResponseForceGang Message	351
Table 5-184 GangStatus Message	351
Table 5-185 CommandGangStatus Message	351
Table 5-186 GangQuFilter Element	352
Table 5-187 ResponseGangStatus Message	352
Table 5-188 GangInfo Element	353
Table 6-1 Template for Input Resources	356
Table 6-2 Generic Input Resources	357
Table 6-3 Template for Output Resources	357
Table 6-4 Approval – Input Resources	358
Table 6-5 Approval – Output Resources	358
Table 6-6 Buffer – Input Resources	358
Table 6-7 Buffer – Output Resources	359
Table 6-8 Combine – Input Resources	359
Table 6-9 Combine – Output Resources	359
Table 6-10 Delivery – Input Resources	359
Table 6-11 Delivery – Output Resources	360
Table 6-12 ManualLabor – Input Resources	360
Table 6-13 ManualLabor – Output Resources	360
Table 6-14 QualityControl – Input Resources	360
Table 6-15 QualityControl – Output Resources	361
Table 6-16 ResourceDefinition – Input Resources	361
Table 6-17 ResourceDefinition – Output Resources	361
Table 6-18 Split – Input Resources	361
Table 6-19 Split – Output Resources	361
Table 6-20 Verification – Input Resources	362
Table 6-21 Verification – Output Resources	362
Table 6-22 AssetListCreation – Input Resources	363
Table 6-23 AssetListCreation – Output Resources	363
Table 6-24 Bending – Input Resources	363
Table 6-25 Bending – Output Resources	363
Table 6-26 ColorCorrection – Input Resources	364
Table 6-27 ColorCorrection – Output Resources	364
Table 6-28 ColorSpaceConversion – Input Resources	365
Table 6-29 ColorSpaceConversion – Output Resources	365
Table 6-30 ContactCopying – Input Resources	365
Table 6-31 ContactCopying – Output Resources	365
Table 6-32 ContoneCalibration – Input Resources	366
Table 6-33 ContoneCalibration – Output Resources	366
Table 6-34 CylinderLayoutPreparation – Input Resources	366
Table 6-35 CylinderLayoutPreparation – Output Resources	366
Table 6-36 DieDesign – Input Resources	367
Table 6-37 DieDesign – Output Resources	367
Table 6-38 DieLayoutProduction – Input Resources	367

Table 6-39 DieLayoutProduction – Output Resources	368
Table 6-40 DigitalDelivery – Input Resources	370
Table 6-41 DigitalDelivery – Output Resources	370
Table 6-42 ImageEnhancement – Input Resources	371
Table 6-43 ImageEnhancement – Output Resources	371
Table 6-44 ImageReplacement – Input Resources.....	371
Table 6-45 ImageReplacement – Output Resources	371
Table 6-46 ImageSetting – Input Resources	372
Table 6-47 ImageSetting – Output Resources	372
Table 6-48 Imposition – Input Resources	372
Table 6-49 Imposition – Output Resources	373
Table 6-50 Glossary for Automated Imposition	373
Table 6-51 Variables for Automated Imposition	377
Table 6-52 InkZoneCalculation – Input Resources	385
Table 6-53 InkZoneCalculation – Output Resources	386
Table 6-54 Interpreting – Input Resources	386
Table 6-55 Interpreting – Output Resources	386
Table 6-56 LayoutElementProduction – Input Resources	387
Table 6-57 LayoutElementProduction – Output Resources	387
Table 6-58 LayoutPreparation – Input Resources.....	387
Table 6-59 LayoutPreparation – Output Resources.....	387
Table 6-60 LayoutShifting – Input Resources	388
Table 6-61 LayoutShifting – Output Resources	388
Table 6-62 PageAssigning – Input Resources	388
Table 6-63 PageAssigning – Output Resources	388
Table 6-64 PDFToPSConversion – Input Resources	389
Table 6-65 PDFToPSConversion – Output Resources	389
Table 6-66 PDLCreation – Input Resources	389
Table 6-67 PDLCreation – Output Resources	389
Table 6-68 Preflight – Input Resources	390
Table 6-69 Preflight – Output Resources	390
Table 6-70 PreviewGeneration – Input Resources	392
Table 6-71 PreviewGeneration – Output Resources	392
Table 6-72 PSToPDFConversion – Input Resources	392
Table 6-73 PSToPDFConversion – Output Resources	393
Table 6-74 RasterReading – Input Resources	393
Table 6-75 RasterReading – Output Resources	393
Table 6-76 Rendering – Input Resources	393
Table 6-77 Rendering – Output Resources.....	394
Table 6-78 Scanning – Input Resources	395
Table 6-79 Scanning – Output Resources	395
Table 6-80 Screening – Input Resources	395
Table 6-81 Screening – Output Resources	396
Table 6-82 Separation – Input Resources	396
Table 6-83 Separation – Output Resources	396
Table 6-84 SheetOptimizing – Input Resources	397
Table 6-85 SheetOptimizing – Output Resources	397
Table 6-86 Stripping – Input Resources	398
Table 6-87 Stripping – Output Resources	398
Table 6-88 Tiling – Input Resources	399

Table 6-89 Tiling – Output Resources	400
Table 6-90 Trapping – Input Resources	400
Table 6-91 Trapping – Output Resources	400
Table 6-92 ConventionalPrinting – Input Resources	401
Table 6-93 ConventionalPrinting – Output Resources	403
Table 6-94 DigitalPrinting – Input Resources	404
Table 6-95 DigitalPrinting – Output Resources	405
Table 6-96 Varnishing – Input Resources	405
Table 6-97 Varnishing – Output Resources	405
Table 6-98 BlockPreparation – Input Resources	406
Table 6-99 BlockPreparation – Output Resources	406
Table 6-100 BoxFolding – Input Resources	407
Table 6-101 BoxFolding – Output Resources	407
Table 6-102 BoxPacking – Input Resources	407
Table 6-103 BoxPacking – Output Resources	407
Table 6-104 Bundling – Input Resources	408
Table 6-105 Bundling – Output Resources	408
Table 6-106 CaseMaking – Input Resources	409
Table 6-107 CaseMaking – Output Resources	409
Table 6-108 CasingIn – Input Resources	409
Table 6-109 CasingIn – Output Resources	409
Table 6-110 ChannelBinding – Input Resources	410
Table 6-111 ChannelBinding – Output Resources	410
Table 6-112 CoilBinding – Input Resources	410
Table 6-113 CoilBinding – Output Resources	411
Table 6-114 Collecting – Input Resources	411
Table 6-115 Collecting – Output Resources	411
Table 6-116 CoverApplication – Input Resources	412
Table 6-117 CoverApplication – Output Resources	412
Table 6-118 Creasing – Input Resources	412
Table 6-119 Creasing – Output Resources	412
Table 6-120 Cutting – Input Resources	413
Table 6-121 Cutting – Output Resources	413
Table 6-122 DieMaking – Input Resources	413
Table 6-123 DieMaking – Output Resources	413
Table 6-124 Embossing – Input Resources	414
Table 6-125 Embossing – Output Resources	414
Table 6-126 EndSheetGluing – Input Resources	414
Table 6-127 EndSheetGluing – Output Resources	415
Table 6-128 Feeding – Input Resources	416
Table 6-129 Feeding – Output Resources	416
Table 6-130 Folding – Input Resources	416
Table 6-131 Folding – Output Resources	416
Table 6-132 Gathering – Input Resources	417
Table 6-133 Gathering – Output Resources	417
Table 6-134 Gluing – Input Resources	417
Table 6-135 Gluing – Output Resources	417
Table 6-136 HeadBandApplication – Input Resources	418
Table 6-137 HeadBandApplication – Output Resources	418
Table 6-138 HoleMaking – Input Resources	418

Table 6-139 HoleMaking – Output Resources	418
Table 6-140 Inserting – Input Resources	418
Table 6-141 Inserting – Output Resources	419
Table 6-142 Jacketing – Input Resources	419
Table 6-143 Jacketing – Output Resources	419
Table 6-144 Labeling – Input Resources	419
Table 6-145 Labeling – Output Resources	420
Table 6-146 Laminating – Input Resources	420
Table 6-147 Laminating – Output Resources	420
Table 6-148 LooseBinding – Input Resources	421
Table 6-149 LooseBinding – Output Resources	422
Table 6-150 Palletizing – Input Resources	422
Table 6-151 Palletizing – Output Resources	423
Table 6-152 Perforating – Input Resources	423
Table 6-153 Perforating – Output Resources	423
Table 6-154 PlasticCombBinding – Input Resources.....	423
Table 6-155 PlasticCombBinding – Output Resources.....	424
Table 6-156 PrintRolling – Input Resources	424
Table 6-157 PrintRolling – Output Resources	424
Table 6-158 RingBinding – Input Resources.....	425
Table 6-159 RingBinding – Output Resources.....	425
Table 6-160 ShapeCutting – Input Resources	425
Table 6-161 ShapeCutting – Output Resources	426
Table 6-162 ShapeDefProduction – Input Resources	426
Table 6-163 ShapeDefProduction – Output Resources	426
Table 6-164 Shrinking – Input Resources	426
Table 6-165 Shrinking – Output Resources	427
Table 6-166 SpinePreparation – Input Resources	427
Table 6-167 SpinePreparation – Output Resources	427
Table 6-168 SpineTaping – Input Resources	427
Table 6-169 SpineTaping – Output Resources	427
Table 6-170 Stacking – Input Resources	428
Table 6-171 Stacking – Output Resources	428
Table 6-172 StaticBlocking – Input Resources	428
Table 6-173 StaticBlocking – Output Resources	428
Table 6-174 Stitching – Input Resources	428
Table 6-175 Stitching – Output Resources	429
Table 6-176 Strapping – Input Resources	429
Table 6-177 Strapping – Output Resources	429
Table 6-178 StripBinding – Input Resources.....	430
Table 6-179 StripBinding – Output Resources.....	430
Table 6-180 ThreadSealing – Input Resources	430
Table 6-181 ThreadSealing – Output Resources	430
Table 6-182 ThreadSewing – Input Resources	431
Table 6-183 ThreadSewing – Output Resources	431
Table 6-184 Trimming – Input Resources	431
Table 6-185 Trimming – Output Resources	431
Table 6-186 WebInlineFinishing – Input Resources	431
Table 6-187 WebInlineFinishing – Output Resources	432
Table 6-188 Winding – Input Resources	432

Table 6-189 Winding – Output Resources	432
Table 6-190 WireCombBinding – Input Resources	433
Table 6-191 WireCombBinding – Output Resources	433
Table 6-192 Wrapping – Input Resources	433
Table 6-193 Wrapping – Output Resources	433
Table 7-1 Product Element	439
Table 7-2 ProductType Attribute Values	440
Table 7-3 Intent Element	441
Table 7-4 Product Intent – Input Resources Intent Elements	442
Table 7-5 Product Intent – Output Resources	443
Table 7-6 Template for Intent Resources	444
Table 7-7 Abstract Span Element.....	445
Table 7-8 List of Span Elements	445
Table 7-9 DurationSpan Element	446
Table 7-10 EnumerationSpan Element	446
Table 7-11 IntegerSpan Element.....	447
Table 7-12 NameSpan Element	447
Table 7-13 NumberSpan Element.....	448
Table 7-14 OptionSpan Element.....	448
Table 7-15 ShapeSpan Element	448
Table 7-16 StringSpan Element	449
Table 7-17 TimeSpan Element.....	449
Table 7-18 XYPairSpan Element	450
Table 7-19 ArtDeliveryIntent Resource	450
Table 7-20 ArtDelivery Element.....	453
Table 7-21 AssemblingIntent Intent Element	456
Table 7-22 Assemble Element	457
Table 7-23 BindIn Element.....	457
Table 7-24 BlowIn Element	457
Table 7-25 StickOn Element	458
Table 7-26 BindingIntent Resource Intent Element	458
Table 7-27 BindingType Attribute Values	461
Table 7-28 BindList Element	461
Table 7-29 BindItem Element.....	462
Table 7-30 ChannelBinding Element.....	463
Table 7-31 CoilBinding Element.....	463
Table 7-32 EdgeGluing Element	463
Table 7-33 HardCoverBinding Element	464
Table 7-34 LooseBinding Element	467
Table 7-35 PlasticCombBinding Element.....	468
Table 7-36 RingBinding Element.....	468
Table 7-37 SaddleStitching Element	470
Table 7-38 SideSewing Element.....	470
Table 7-39 SideStitching Element	470
Table 7-40 SoftCoverBinding Element	470
Table 7-41 StripBinding Element.....	472
Table 7-42 Tabs Element	472
Table 7-43 Tape Element	473
Table 7-44 ThreadSealing Element.....	473
Table 7-45 ThreadSewing Element	473

Table 7-46 WireCombBinding Element	473
Table 7-47 ColorIntent Resource Intent Element	475
Table 7-48 SurfaceColor Element	479
Table 7-49 ColorStandard Attribute Values	480
Table 7-50 ColorsUsed Element	481
Table 7-51 ContentCheckIntent Intent Element	481
Table 7-52 ProofItem Element	482
Table 7-53 DeliveryIntent Resource Intent Element.....	483
Table 7-54 DropIntent Element	486
Table 7-55 DropItemIntent Element	489
Table 7-56 EmbossingIntent Resource Intent Element	490
Table 7-57 EmbossingItem Element	490
Table 7-58 FoldingIntent Resource Intent Element	492
Table 7-59 HoleMakingIntent Resource Intent Element	493
Table 7-60 InsertingIntent Resource	494
Table 7-61 InsertList Element	495
Table 7-62 Insert Element.....	495
Table 7-63 LaminatingIntent Resource Intent Element	496
Table 7-64 LayoutIntent Resource Intent Element	497
Table 7-65 MediaIntent Resource Intent Element	500
Table 7-66 PackingIntent Resource	508
Table 7-67 ProductionIntent Resource Intent Element	510
Table 7-68 ProofingIntent Resource Intent Element	511
Table 7-69 ProofItem Element	511
Table 7-70 PublishingIntent Resource Intent Element.....	513
Table 7-71 ScreeningIntent Resource	513
Table 7-72 ShapeCuttingIntent Resource Intent Element	514
Table 7-73 ShapeCut Element	514
Table 7-74 VariableIntent Intent Element	515
Table 8-1 Resource Element	517
Table 8-2 Location Element	520
Table 8-3 ApprovalParams Resource	521
Table 8-4 ApprovalPerson Element	521
Table 8-5 ApprovalSuccess Resource	522
Table 8-6 ApprovalDetails Element Resource	522
Table 8-7 Assembly Resource	523
Table 8-8 AssemblySection Element	525
Table 8-9 PageAssignedList Element	526
Table 8-10 AssetListCreationParams Resource	527
Table 8-11 BendingParams Resource	528
Table 8-12 BinderySignature Resource	529
Table 8-13 SignatureCell Element	539
Table 8-14 BlockPreparationParams Resource	544
Table 8-15 BoxFoldingParams Resource	545
Table 8-16 BoxFoldAction Element	546
Table 8-17 Action Attribute Values	546
Table 8-18 BoxPackingParams Resource	550
Table 8-19 BufferParams Resource	552
Table 8-20 Bundle Resource	552
Table 8-21 BundleItem Element	553

Table 8-22 BundlingParams Resource	555
Table 8-23 ByteMap Resource.....	556
Table 8-24 Band Element.....	557
Table 8-25 PixelColorant Element.....	557
Table 8-26 CaseMakingParams Resource	558
Table 8-27 CasingInParams Resource	559
Table 8-28 ChannelBindingParams Resource	560
Table 8-29 CoilBindingParams Resource	561
Table 8-30 CollectingParams Resource	562
Table 8-31 Color Resource	564
Table 8-32 DeviceNColor Element.....	569
Table 8-33 Diecutting Data (DDES3)	569
Table 8-34 PrintConditionColor Element	570
Table 8-35 ColorantControl Resource	575
Table 8-36 ColorantConvertProcess Element.....	580
Table 8-37 ColorantOrder Element	580
Table 8-38 ColorantParams Element.....	580
Table 8-39 DeviceColorantOrder Element	580
Table 8-40 ColorSpaceSubstitute Element	580
Table 8-41 Sample output for different values of ProcessColorModel, ColorantParams, ColorantOrder, ColorantControlLink and DeviceColorantOrder Elements	581
Table 8-42 Sample output for different values of ProcessColorModel, ColorantParams, ColorantOrder, ColorantControlLink and DeviceColorantOrder Elements	582
Table 8-43 DeviceNSpace Element	582
Table 8-44 ColorCorrectionParams Resource	584
Table 8-45 ColorPool Resource	585
Table 8-46 ColorSpaceConversionParams Resource	585
Table 8-47 Glossary – Component	588
Table 8-48 Component Resource	589
Table 8-49 ProductType Attribute Values	593
Table 8-50 Contact Resource	595
Table 8-51 ComChannel Element	597
Table 8-52 ChannelTypeDetails Attribute – predefined values for certain ChannelType values	598
Table 8-53 Company Element.....	598
Table 8-54 Person Element	598
Table 8-55 ContactCopyParams Resource	599
Table 8-56 ContentList Resource	600
Table 8-57 ContentData Resource Element	600
Table 8-58 ContentType Attribute Values	603
Table 8-59 BarcodeProductionParams Element	603
Table 8-60 ContentMetadata Element	604
Table 8-61 PositionObj Element.....	604
Table 8-62 ConventionalPrintingParams Resource	608
Table 8-63 CoverApplicationParams Resource	611
Table 8-64 Score Element	611
Table 8-65 CreasingParams Resource	613
Table 8-66 CustomerInfo Resource	614
Table 8-67 CutBlock Resource	615
Table 8-68 CutMark Resource	616

Table 8-69 Cut mark types as specified by CutMark/@MarkType	617
Table 8-70 CuttingParams Resource	618
Table 8-71 CutBlock Element	619
Table 8-72 CylinderLayout Resource	619
Table 8-73 CylinderPosition Element	620
Table 8-74 CylinderLayoutPreparationParams Resource	623
Table 8-75 DeliveryParams Resource	624
Table 8-76 Drop Element	628
Table 8-77 DroplItem Element	629
Table 8-78 DevelopingParams Resource	631
Table 8-79 Device Resource	632
Table 8-80 IconList Element	635
Table 8-81 Icon Element	635
Table 8-82 Module Element	636
Table 8-83 DieLayout Resource	637
Table 8-84 RuleLength Element	637
Table 8-85 Station Element	638
Table 8-86 DieLayoutProductionParams Resource	638
Table 8-87 RepeatDesc Element	639
Table 8-88 DigitalDeliveryParams Resource	643
Table 8-89 DigitalMedia Resource	644
Table 8-90 DigitalPrintingParams Resource	646
Table 8-91 ElementColorParams Resource	650
Table 8-92 EmbossingParams Resource	652
Table 8-93 Emboss Element	652
Table 8-94 Employee Resource	654
Table 8-95 EndSheetGluingParams Resource	655
Table 8-96 EndSheet Element	655
Table 8-97 ExposedMedia Resource	657
Table 8-98 ExternalImpositionTemplate Resource	658
Table 8-99 FeedingParams Resource	659
Table 8-100 Feeder Element	659
Table 8-101 FeederQualityParams Element	660
Table 8-102 CollatingItem Element	661
Table 8-103 FileSpec Resource	663
Table 8-104 ResourceUsage Attribute Values	668
Table 8-105 Container Element	669
Table 8-106 Disposition Element	670
Table 8-107 FileAlias Element	670
Table 8-108 FoldingParams Resource	672
Table 8-109 FontParams Resource	676
Table 8-110 FontPolicy Resource	677
Table 8-111 GatheringParams Resource	678
Table 8-112 GlueApplication Resource	679
Table 8-113 GluingParams Resource	680
Table 8-114 Glue Element	680
Table 8-115 HeadBandApplicationParams Resource	680
Table 8-116 HoleList Resource	682
Table 8-117 HoleMakingParams Resource	682
Table 8-118 IdentificationField Resource	685

Table 8-119 EncodingDetails Attribute Values	687
Table 8-120 BarcodeDetails Element	688
Table 8-121 ExtraValues Element	689
Table 8-122 Usage of Barcode Attributes for Certain Barcode Types	690
Table 8-123 BarcodeVersion Values – for HIBC_DATAMATRIX	690
Table 8-124 BarcodeVersion Values – for QR barcodes	691
Table 8-125 ImageCompressionParams Resource	692
Table 8-126 ImageCompression Element	692
Table 8-127 CCITTFaxParams Element	695
Table 8-128 DCTParams Element	695
Table 8-129 SourceCSs Attribute Values	696
Table 8-130 FlateParams Element	697
Table 8-131 JBIG2Params Element	697
Table 8-132 JPEG2000Params Element	698
Table 8-133 LZWParams Element	698
Table 8-134 ImageEnhancementParams Resource	699
Table 8-135 ImageEnhancementOp Element	699
Table 8-136 ImageReplacementParams Resource	700
Table 8-137 ImageSetterParams Resource	702
Table 8-138 Sides Attribute Values	704
Table 8-139 Ink Resource	704
Table 8-140 InkZoneCalculationParams Resource	706
Table 8-141 InkZoneProfile Resource	707
Table 8-142 InsertingParams Resource	708
Table 8-143 Location of Inserts	709
Table 8-144 InterpretingParams Resource	710
Table 8-145 InterpretingDetails Element	713
Table 8-146 PDFInterpretingParams Element	713
Table 8-147 OCGControl Element	715
Table 8-148 ReferenceXObjParams Element	715
Table 8-149 PDLResourceAlias Element	716
Table 8-150 JacketingParams Resource	717
Table 8-151 LabelingParams Resource	718
Table 8-152 LaminatingParams Resource	719
Table 8-153 Layout Resource	720
Table 8-154 ColorControlStrip Element	726
Table 8-155 CIELABMeasuringField Element	727
Table 8-156 DensityMeasuringField Resource	728
Table 8-157 DeviceMark Element	729
Table 8-158 LayerList Element	732
Table 8-159 LayerDetails Element	733
Table 8-160 LogicalStackParams Element	733
Table 8-161 Stack Element	733
Table 8-162 PageCondition Element	734
Table 8-163 SheetCondition Element	736
Table 8-164 Abstract PlacedObject Element	737
Table 8-165 ContentObject Element	742
Table 8-166 MarkObject Element	748
Table 8-167 CutMark Resource	754
Table 8-168 Cut mark types as specified by CutMark/@MarkType	755

Table 8-169 FillMark Element	755
Table 8-170 MarkActivation Element	756
Table 8-171 DynamicField Element	757
Table 8-172 Example (1) of Ord Attribute in PlacedObject Elements	761
Table 8-173 Example (2) of Ord Attribute in PlacedObject Elements	762
Table 8-174 LayoutElement Resource.....	763
Table 8-175 ElementType Attribute Values	767
Table 8-176 Dependencies Element	767
Table 8-177 LayoutElementProductionParams Resource	768
Table 8-178 LayoutElementPart Element	770
Table 8-179 BarcodeProductionParams Element	771
Table 8-180 PositionObj Element.....	772
Table 8-181 LayoutPreparationParams Resource	778
Table 8-182 FrontMarkList Attribute Values	786
Table 8-183 PageDistributionScheme Attribute Values	786
Table 8-184 PageCell Element	787
Table 8-185 ImageShift Element.....	789
Table 8-186 LayoutShift Resource	793
Table 8-187 ShiftPoint Element	793
Table 8-188 LooseBindingParams Resource	795
Table 8-189 ChannelBindingDetails Element	797
Table 8-190 CoilBindingDetails Element	798
Table 8-191 CombBindingDetails Element	798
Table 8-192 RingBindingDetails Element	799
Table 8-193 ManualLaborParams Resource	800
Table 8-194 Media Resource	801
Table 8-195 MediaTypeDetails Attribute Values	812
Table 8-196 TabDimensions Element	815
Table 8-197 MiscConsumable Resource	821
Table 8-198 NodeInfo Resource	821
Table 8-199 NodeStatus Attribute Values	824
Table 8-200 PageAssignParams Resource	825
Table 8-201 PageList Resource	826
Table 8-202 PageData Element	828
Table 8-203 PageElement Element	831
Table 8-204 Pallet Resource	832
Table 8-205 PalletizingParams Resource	832
Table 8-206 PDFToPSConversionParams Resource	833
Table 8-207 PDLCreationParams Resource	836
Table 8-208 FontParams Element	837
Table 8-209 PDFToPSConversionParams Resource PSCreationDetails Element	838
Table 8-210 PSToPDFConversionParams Resource PDFCreationDetails Element	840
Table 8-211 AdvancedParams Element	842
Table 8-212 PDFXParams Element	844
Table 8-213 ThinPDFParams Element	846
Table 8-214 PDLResourceAlias Resource	847
Table 8-215 PerforatingParams Resource	847
Table 8-216 PlasticCombBindingParams Resource	848
Table 8-217 PreflightParams Resource	849
Table 8-218 PreflightTest Element	850

Table 8-219 PreflightAction Element.....	850
Table 8-220 BasicPreflightTest Element	851
Table 8-221 PreflightArgument Element	852
Table 8-222 BoxArgument Element	852
Table 8-223 Box Attribute Values	853
Table 8-224 BoxToBoxDifference Element	853
Table 8-225 PreflightReport Resource	854
Table 8-226 PreflightCheck Element.....	855
Table 8-227 PRItem Element.....	856
Table 8-228 PRError Element.....	856
Table 8-229 PRGroup Element.....	857
Table 8-230 Abstract PRGroupOccurrenceBase Element	859
Table 8-231 List of PRGroupOccurrenceBase Elements	859
Table 8-232 ArgumentValue Element.....	859
Table 8-233 PRGroupOccurrence Element	859
Table 8-234 StringListValue Element.....	860
Table 8-235 PROccurrence Element	860
Table 8-236 PreflightReportRulePool Resource	860
Table 8-237 PRRule Element.....	861
Table 8-238 PRRuleAttr Element.....	861
Table 8-239 ReportAttr Attribute Values	861
Table 8-240 Contingent Report Behavior.....	862
Table 8-241 Preview Resource	863
Table 8-242 PreviewGenerationParams Resource	865
Table 8-243 PrintCondition Resource	867
Table 8-244 PrintRollingParams Resource	868
Table 8-245 ProductionPath Resource	869
Table 8-246 FolderSuperstructureWebPath Element	869
Table 8-247 PostPressComponentPath Element.....	869
Table 8-248 PrintingUnitWebPath Element.....	869
Table 8-249 PSToPDFConversionParams Resource	870
Table 8-250 AdvancedParams Element	872
Table 8-251 PDFXParams Element	874
Table 8-252 PDFXOutputIntentProfile Attribute Values	876
Table 8-253 ThinPDFParams Element	876
Table 8-254 QualityControlParams Resource	877
Table 8-255 BindingQualityParams Element	877
Table 8-256 QualityControlResult Resource	878
Table 8-257 QualityMeasurement Element	878
Table 8-258 BindingQualityMeasurement Element	878
Table 8-259 RasterReadingParams Resource	879
Table 8-260 RegisterMark Resource	880
Table 8-261 RenderingParams Resource	881
Table 8-262 TIFFFormatParams Element	882
Table 8-263 TIFFtag Element	884
Table 8-264 TIFFEmbeddedFile Element	884
Table 8-265 ResourceDefinitionParams Resource	885
Table 8-266 ResourceParam Element.....	885
Table 8-267 RingBindingParams Resource	886
Table 8-268 RollStand Resource	887

Table 8-269 RunList Resource	889
Table 8-270 ByteMap Element	899
Table 8-271 Band Element	900
Table 8-272 ElementType Attribute Values	900
Table 8-273 ScanParams Resource	903
Table 8-274 ScavengerArea Resource	905
Table 8-275 ScreeningParams Resource	905
Table 8-276 SeparationControlParams Resource	906
Table 8-277 Shape Resource	906
Table 8-278 ShapeCuttingParams Resource	908
Table 8-279 ShapeDef Resource	909
Table 8-280 CutLines Element.....	910
Table 8-281 ShapeDefProductionParams Resource	910
Table 8-282 ObjectModel Element	911
Table 8-283 ShapeTemplate Element	911
Table 8-284 ShapeDimension Element	912
Table 8-285 SheetOptimizingParams Resource	914
Table 8-286 GangElement Element.....	914
Table 8-287 SeparationListBack Element.....	917
Table 8-288 SeparationListFront Element.....	918
Table 8-289 ShrinkingParams Resource	918
Table 8-290 SpinePreparationParams Resource	919
Table 8-291 Operations Attribute Values	919
Table 8-292 SpineTapingParams Resource	920
Table 8-293 Parameters in Stacking	922
Table 8-294 StackingParams Resource	924
Table 8-295 StaticBlockingParams Resource	925
Table 8-296 StitchingParams Resource	928
Table 8-297 Strap Resource	930
Table 8-298 StrappingParams Resource	930
Table 8-299 StripBindingParams Resource	931
Table 8-300 StrippingParams Resource	932
Table 8-301 ExternallImpositionTemplate Element	937
Table 8-302 Position Element	937
Table 8-303 StripCellParams Element	938
Table 8-304 StripMark Element	941
Table 8-305 MarkName Attribute Values	944
Table 8-306 MarkSide Attribute Values	945
Table 8-307 ThreadSealingParams Resource	946
Table 8-308 ThreadSewingParams Resource	947
Table 8-309 Tile Resource	949
Table 8-310 Tool Resource	950
Table 8-311 TransferCurve Resource	951
Table 8-312 TransferCurvePool Resource	951
Table 8-313 TransferCurveSet Element	952
Table 8-314 TransferFunctionControl Resource	952
Table 8-315 TrappingDetails Resource	953
Table 8-316 TrappingOrder Element.....	954
Table 8-317 TrappingParams Resource	955
Table 8-318 ColorantZoneDetails Element	958

Table 8-319 TrapRegion Resource	959
Table 8-320 TrimmingParams Resource	960
Table 8-321 UsageCounter Resource	960
Table 8-322 VarnishingParams Resource	962
Table 8-323 VerificationParams Resource	963
Table 8-324 WebInlineFinishingParams Resource	964
Table 8-325 FolderProduction Element	964
Table 8-326 WindingParams Resource	964
Table 8-327 WireCombBindingParams Resource	965
Table 8-328 WrappingParams Resource	966
Table 9-1 DeviceCap Element	971
Table 9-2 ActionPool Element	975
Table 9-3 Action Element	975
Table 9-4 DevCapPool Element	975
Table 9-5 StatePool Element	975
Table 9-6 ModulePool Element	976
Table 9-7 ModuleCap Element	976
Table 9-8 DevCaps Element	977
Table 9-9 Loc Element	979
Table 9-10 DevCap Element	980
Table 9-11 Abstract State Element	983
Table 9-12 ListType Attribute Values	985
Table 9-13 List of State Elements	986
Table 9-14 BooleanState Element	988
Table 9-15 ValueLoc Element	989
Table 9-16 DateTimeState Element	990
Table 9-17 DurationState Element	991
Table 9-18 ElementState Element	993
Table 9-19 EnumerationState Element	994
Table 9-20 IntegerState Element	995
Table 9-21 MatrixState Element	999
Table 9-22 MatrixState/Value Element	1000
Table 9-23 NameState Element	1001
Table 9-24 NumberState Element	1002
Table 9-25 PDFPathState Element	1004
Table 9-26 PDFPathState/Value Element	1005
Table 9-27 RectangleState Element	1005
Table 9-28 ShapeState Element	1007
Table 9-29 StringState Element	1009
Table 9-30 StringState/Value Element	1011
Table 9-31 XYPairState Element	1012
Table 9-32 DisplayGroupPool Element	1014
Table 9-33 DisplayGroup Element	1014
Table 9-34 FeaturePool Element	1015
Table 9-35 MacroPool Element	1015
Table 9-36 macro Element	1016
Table 9-37 choice Element	1017
Table 9-38 otherwise Element	1017
Table 9-39 when Element	1017
Table 9-40 set Element	1017

Table 9-41 FeatureAttribute Element	1017
Table 9-42 call Element.....	1018
Table 9-43 Performance Element	1018
Table 9-44 TestPool Element	1019
Table 9-45 Test Element	1019
Table 9-46 List of Term Elements	1020
Table 9-47 and Element	1022
Table 9-48 or Element	1023
Table 9-49 xor Element	1023
Table 9-50 not Element	1023
Table 9-51 TestRef Element	1023
Table 9-52 Abstract Evaluation Element	1025
Table 9-53 List of Evaluation Elements	1025
Table 9-54 Mapping of Evaluation Element to State Element	1026
Table 9-55 BooleanEvaluation Element	1028
Table 9-56 DateTimeEvaluation Element	1028
Table 9-57 DurationEvaluation Element	1028
Table 9-58 EnumerationEvaluation Element.....	1029
Table 9-59 IntegerEvaluation Element	1029
Table 9-60 IsPresentEvaluation Element	1029
Table 9-61 MatrixEvaluation Element	1030
Table 9-62 MatrixEvaluation/Value Element	1030
Table 9-63 NameEvaluation Element.....	1030
Table 9-64 NumberEvaluation Element	1031
Table 9-65 PDFPathEvaluation Element.....	1031
Table 9-66 PDFPathEvaluation/Value Element.....	1031
Table 9-67 RectangleEvaluation Element	1031
Table 9-68 ShapeEvaluation Element	1032
Table 9-69 StringEvaluation Element	1032
Table 9-70 StringEvaluation/Value Element	1033
Table 9-71 XYPairEvaluation Element	1033
Table 9-72 Object Classes for a Document	1043
Table 9-73 Properties Implemented by each Class of Object	1043
Table 9-74 Mapping between property types (in the preflight spec) and evaluations	1045
Table 9-75 List of Properties Categories.....	1046
Table 9-76 Annotation Properties.....	1046
Table 9-77 AnnotationType Attribute Values	1047
Table 9-78 Box Properties.....	1047
Table 9-79 Class Properties.....	1047
Table 9-80 ClassName Attribute Values	1048
Table 9-81 PropertyList Attribute Values	1048
Table 9-82 Colorant Properties	1048
Table 9-83 Document Properties	1049
Table 9-84 Fill Properties	1053
Table 9-85 FillColorType Attribute Values	1053
Table 9-86 Font Properties.....	1054
Table 9-87 FontType Attribute Values	1054
Table 9-88 Graphic Properties	1055
Table 9-89 Image Properties.....	1057
Table 9-90 Logical Properties	1059

Table 9-91 PageBox Properties	1059
Table 9-92 Pages Properties.....	1059
Table 9-93 PDLObject Properties	1061
Table 9-94 Reference Properties	1061
Table 9-95 Shading Properties.....	1062
Table 9-96 Stroke Properties	1062
Table 9-97 Text Properties	1063
Table 9-98 Vector Properties.....	1063
Table 10-1 Address Element.....	1065
Table 10-2 ApprovalPerson Element	1065
Table 10-3 AutomatedOverPrintParams Element	1066
Table 10-4 BarcodeCompParams Element	1067
Table 10-5 BarcodeReproParams Element	1067
Table 10-6 ColorantAlias Element.....	1068
Table 10-7 ColorCorrectionOp Element	1069
Table 10-8 ColorMeasurementConditions Resource	1071
Table 10-9 ColorSpaceConversionOp Element	1072
Table 10-10 SourceCS Attribute Values	1076
Table 10-11 Mapping of SourceCS enumeration values to color spaces in the most common input file formats	1077
Table 10-12 ComChannel Element	1079
Table 10-13 ChannelTypeDetails Attribute – predefined values for certain ChannelType values ..	1080
Table 10-14 ConvertingConfig Element	1081
Table 10-15 CostCenter Element	1081
Table 10-16 Crease Element	1082
Table 10-17 Cut Element	1083
Table 10-18 DeviceMark Element	1084
Table 10-19 DeviceNSpace Element	1088
Table 10-20 Disjointing Element	1088
Table 10-21 Disposition Element	1090
Table 10-22 FitPolicy Element	1091
Table 10-23 Fold Element	1092
Table 10-24 GlueLine Element	1093
Table 10-25 HolePattern Element.....	1095
Table 10-26 HoleLine Element.....	1096
Table 10-27 InsertSheet Element	1097
Table 10-28 SheetUsage Attribute Values	1099
Table 10-29 JobField Element	1101
Table 10-30 MarkColor Element	1102
Table 10-31 MediaLayers Element	1102
Table 10-32 MetadataMap Element	1103
Table 10-33 Expr Element	1105
Table 10-34 MetadataMap: Setting Attributes	1106
Table 10-35 MISDetails Element	1108
Table 10-36 ObjectResolution Element	1109
Table 10-37 OCGControl Element	1110
Table 10-38 Perforate Element	1110
Table 10-39 Person Element	1112
Table 10-40 RefAnchor Element	1112

Table 10-41 RegisterRibbon Element	1113
Table 10-42 ScreenSelector Element	1114
Table 10-43 SeparationSpec Element	1117
Table 11-1 Actions generated when a dynamic-pipe buffer passes various levels	1124
Table 11-2 Event Sequence in Digital Finishing	1125
Table 11-3 XJMF Bootstrapping steps	1134
Table 11-4 MIME Content-Types	1136
Table 11-5 Modifying Job Parameters.....	1140
Table A-1 Anchor Enumeration Values	1156
Table A-2 JDFJMFileVersion Enumeration Values	1157
Table A-3 NamedColor Enumeration Values.....	1157
Table A-4 Orientation Enumeration Values	1157
Table A-5 Sides Enumeration Values.....	1158
Table A-6 Scope Enumeration Values.....	1158
Table A-7 WorkStyle Enumeration Values	1158
Table A-8 XYRelation Enumeration Values	1159
Table C-1 StatusDetails Mapping for Generic Devices	1165
Table C-2 StatusDetails Mapping for Printing Devices	1168
Table C-3 StatusDetails Mapping for Postpress Devices	1170
Table C-4 ModuleType Attribute Values for Conventional Printing Devices	1170
Table C-5 ModuleType Attribute Values for Postpress.....	1171
Table C-6 ModuleType Attribute Values for DigitalPrinting	1172
Table C-7 ModuleType Attribute Values for PrintingUnitWebPath Modules of Web Printing Devices	1172
Table C-8 ModuleType Attribute Values for FolderSuperstructureWebPath Modules of Web Print- ing Devices	1173
Table C-9 ModuleType Attribute Values for PostPressComponentPath Modules of Web Printing Devices	1174
Table C-10 Abstract NotificationDetails.....	1174
Table C-11 List of NotificationDetails Elements	1175
Table C-12 Barcode Element	1175
Table C-13 FCNKey Element	1175
Table C-14 SystemTimeSet Element	1175
Table C-15 CounterReset Element	1175
Table C-16 Error Element	1176
Table C-17 ErrorData Element	1176
Table C-18 Event Element	1176
Table C-19 Milestone Element	1177
Table C-20 MessageEvents and MilestoneType Values	1178
Table C-21 Input Tray and Output Bin Names	1179
Table D-1 Return codes for JMF XJMF	1183
Table E-1 Attributes for Color Space Adjustment	1187
Table F-1 Conversion Factor from USWeight (lbs) to Weight (g/m2)	1189
Table F-2 Grammage Equivalents for Common (US) Basis Weights	1189
Table F-3 Japanese Media Weight	1191
Table G-1 Media Sizes	1193
Table H-1 MimeType Attribute Values (IANA Registered)	1197
Table H-2 MimeType and File Type Combinations	1199
Table I-1 Predefined variables used in @XXXTemplate and @ShowList	1203
Table J-1 Schemes Names for Binding Orientations	1209

Table J-2 Transformations for each Scheme	1210
Table J-3 Original Diagram	1212
Table J-4 Horizontal Binding Edges	1212
Table J-5 Original Diagram	1213
Table J-6 Vertical Binding Edges	1213
Table J-7 Pagination Diagrams	1214
Table L-1 References	1257
Table M-1 Naming Scheme for Hole Patterns	1271
Table M-2 Hole Details for R2 Series	1272
Table M-3 Hole Details for R3 and R4 Series	1273
Table M-4 Hole Details for R5 and R6 Series	1274
Table M-5 Hole Details for R7 and R11 Series	1276
Table M-6 Hole Details for P, W, C and S Series	1277
Table N-1 Use Cases showingMimeType, URL and Compression Attribute Values	1281
Table N-2 AppOS and OSVersion Examples	1291
Table P-1 Contents of the AbstractResourceUpdate Element	1338
Table P-2 Contents of the StatusPool Element	1338
Table P-3 Contents of the PartStatus Element	1338
Table P-4 Added Element	1341
Table P-5 ChangedAttribute Element	1341
Table P-6 Removed Element	1342
Table P-7 Events Message	1342
Table P-8 NotificationDef Element	1343
Table P-9 KnownControllers Message	1343
Table P-10 ControllerFilter Element	1344
Table P-11 JDFController Element	1344
Table P-12 RepeatMessages Message	1345
Table P-13 MsgFilter Element	1345
Table P-14 NodeInfo Query Message	1347
Table P-15 NodeInfoQuParams Element	1347
Table P-16 NodeInfo Command Message	1347
Table P-17 NodeInfoCmdParams Element	1347
Table P-18 NodeInfoResp Element	1348
Table P-19 KnownJDFServices Message	1349
Table P-20 JDFService Element	1349
Table P-21 Occupation Message	1350
Table P-22 EmployeeDef Element	1350
Table P-23 Occupation Element	1351
Table P-24 Track Message	1352
Table P-25 TrackFilter Element	1352
Table P-26 TrackResult Element	1352
Table P-27 QueueEntryStatus Message	1353
Table P-28 QueueEntryDefList Element	1353
Table P-29 DBDocTemplateLayout – Input Resources	1354
Table P-30 DBDocTemplateLayout – Output Resources	1354
Table P-31 DBTemplateMerging – Input Resources	1354
Table P-32 DBTemplateMerging – Output Resources	1354
Table P-33 FormatConversion – Input Resources	1355
Table P-34 FormatConversion – Output Resources	1355
Table P-35 Ordering – Input Resources	1355

Table P-36 Ordering – Output Resources	1355
Table P-37 Packing – Input Resources.....	1355
Table P-38 Packing – Output Resources	1356
Table P-39 FilmToPlateCopying – Input Resources.....	1356
Table P-40 FilmToPlateCopying – Output Resources.....	1356
Table P-41 PreflightAnalysis Resource	1356
Table P-42 PreflightDetail Element	1357
Table P-43 PreflightInstance Element.....	1358
Table P-44 PreflightInstanceDetail Element.....	1358
Table P-45 PreflightInventory Resource	1359
Table P-46 PreflightProfile Resource	1359
Table P-47 PreflightConstraint Element	1360
Table P-48 Proofing – Input Resources	1361
Table P-49 Proofing – Output Resources	1361
Table P-50 SoftProofing – Input Resources.....	1362
Table P-51 SoftProofing – Output Resources	1362
Table P-52 IDPrinting – Input Resources.....	1362
Table P-53 IDPrinting – Output Resources	1363
Table P-54 Dividing – Input Resources.....	1364
Table P-55 Dividing – Output Resources	1364
Table P-56 LongitudinalRibbonOperations – Input Resources	1364
Table P-57 LongitudinalRibbonOperations – Output Resources	1364
Table P-58 Numbering – Input Resources	1365
Table P-59 Numbering – Output Resources	1365
Table P-60 SaddleStitching – Input Resources.....	1365
Table P-61 SaddleStitching – Output Resources	1365
Table P-62 SideSewing – Input Resources.....	1365
Table P-63 SideSewing – Output Resources.....	1365
Table P-64 AdhesiveBinding Element.....	1366
Table P-65 BookCase Element	1366
Table P-66 Pricing Element.....	1367
Table P-67 Payment Element	1367
Table P-68 CreditCard Element	1367
Table P-69 NumberingIntent Resource Intent Element	1368
Table P-70 NumberItem Element	1369
Table P-71 SizeIntent Resource	1370
Table P-72 AdhesiveBindingParams Resource	1370
Table P-73 BoxApplication Element.....	1371
Table P-74 CustomerMessage Element	1372
Table P-75 DBMergeParams Resource	1372
Table P-76 DBRules Resource	1373
Table P-77 DBSchema Resource	1373
Table P-78 DBSelection Resource.....	1374
Table P-79 DividingParams Resource	1374
Table P-80 FormatConversionParams Resource	1375
Table P-81 IDPrintingParams Resource	1376
Table P-82 OutputBin Attribute Values	1377
Table P-83 Cover Element	1377
Table P-84 IDPFinishing Element	1378
Table P-85 Finishings Attribute Values	1379

Table P-86 IDPFolding Element	1379
Table P-87 IDPHoleMaking Element	1380
Table P-88 IDPLayout Element	1380
Table P-89 IDPStitching Element	1382
Table P-90 IDPTrimming Element	1383
Table P-91 ImageShift Element	1383
Table P-92 JobSheet Element	1384
Table P-93 Signature Element	1387
Table P-94 LongitudinalRibbonOperationParams Resource	1387
Table P-95 LROperation Element	1387
Table P-96 LongGlue Element	1388
Table P-97 LongPerforate Element	1388
Table P-98 MediaSource Resource	1389
Table P-99 NumberingParams Resource	1389
Table P-100 NumberingParam Element	1389
Table P-101 OrderingParams Resource	1390
Table P-102 PackingParams Resource	1391
Table P-103 PlateCopyParams Resource	1392
Table P-104 ProofingParams Resource	1393
Table P-105 DynamicInput Element	1394
Table P-106 SaddleStitchingParams Resource	1394
Table P-107 Sheet Resource	1395
Table P-108 SideSewingParams Resource	1397
Table P-109 Surface Resource	1397

Appendix S List of Examples

Example 1-1 XPath Expression	6
Example 3-1 ResourceLink Structure for a ProcessGroup	70
Example 3-2 Combined Process Node	72
Example 3-3 ResourceLinkPool for Combined Process Node	73
Example 3-4 Complex Combined Process Node	74
Example 3-5 ExposedMedia with Location Elements	88
Example 3-6 Media with Location Elements	88
Example 3-7 Versioned Set Of Plates with Multiple Part Elements	94
Example 3-8 EmployeeLink	111
Example 3-9 PartAmount	115
Example 3-10 MediaLink with Lot	117
Example 3-11 MediaRef to Partitioned Media	119
Example 3-12 Equivalent Inline Media	119
Example 3-13 Invalid Inline Partitioned Media	119
Example 3-14 MediaLink and MediaRef	120
Example 3-15 Inheritance for Subelements of a Partitioned Resource	126
Example 3-16 Partitioned ExposedMedia	126
Example 3-17 Legal Incomplete Partition	127
Example 3-18 Illegal Incomplete Partition	127
Example 3-19 Legal Complete Partition	127
Example 3-20 Illegal Partition	128
Example 3-21 Degenerate Partition	128
Example 3-22 Invalid Degenerate Partition	128
Example 3-23 Partitioned ExposedMedia with Media Subelements	129
Example 3-24 Partitioned ExposedMedia with Incomplete Media Subelements	129
Example 3-25 Partitioned ExposedMedia with Invalid Partitioning of Media Subelements	129
Example 3-26 Partitioned ExposedMedia with MediaRef Subelements	129
Example 3-27 Partitioned ExposedMedia with Invalid MediaRef Subelements	130
Example 3-28 Partitioning with the Identical Element	130
Example 3-29 ResourceLink with Part Element	131
Example 3-30 Partitioning with an Invalid Identical Element	131
Example 3-31 ExposedMedia with Location Elements	145
Example 3-32 Media with Location Elements	145
Example 3-33 Linking to Subsets of Resources	146
Example 3-34 @Amount in an ExposedMediaLink to a Partitioned ExposedMedia	146
Example 3-35 PartUsage in a Partitioned Resource	147
Example 3-36 Explicit Reference of Ordered Partitioned Resources	149
Example 3-37 Implicit Reference of Ordered Partitioned Resources	149
Example 3-38 ResourceAudit: Before Logging	172
Example 3-39 ResourceAudit: Logging of Consumption	172
Example 3-40 ResourceAudit: Logging Changes	173
Example 3-41 Namespaces in XML	175
Example 3-42 Creating Extension Assets	176
Example 3-43 Extending Process Types	176
Example 3-44 Extending NMOKEN Lists	177
Example 4-1 Product Intent Node	184
Example 4-2 Family Tree of Spawned Nodes	200

Example 5-1 Message Type vs Explicit Message Element.....	208
Example 5-2 Query Message	219
Example 5-3 ResumeQueueEntry Command Message	221
Example 5-4 ResumeQueueEntry Response Message	221
Example 5-5 Signal Message	224
Example 5-6 Response Message for Query	226
Example 5-7 Acknowledge Message	228
Example 5-8 Response with Notification Element	234
Example 5-9 KnownDevices Response	237
Example 5-10 KnownMessages Response	244
Example 5-11 Notification Signal	250
Example 5-12 RequestForAuthentication Command.....	258
Example 5-13 RequestForAuthentication Response	258
Example 5-14 Follow up RequestForAuthentication Query	259
Example 5-15 RequestForAuthentication Response from Follow Up Query	259
Example 5-16 Resource Query about Paper	272
Example 5-17 Resource Response about Paper.....	272
Example 5-18 Resource Query about Employees.....	273
Example 5-19 Resource Response about Employees	273
Example 5-20 Resource Signal about Consumed Resources.....	274
Example 5-21 Resource Command: Single Resource is Available	278
Example 5-22 Resource Command: Multiple Resources are Available.....	278
Example 5-23 Resource Query for Consumables	283
Example 5-24 Resource Response about Consumables	283
Example 5-25 Resource Command for Changing Amount	283
Example 5-26 Resource Response for Changing Amount	283
Example 5-27 ResourcePull Command	286
Example 5-28 Status Signal.....	289
Example 5-29 Status Response to Query.....	298
Example 5-30 UpdateJDF Command	302
Example 5-31 AbortQueueEntry Command.....	317
Example 5-32 AbortQueueEntry Response	318
Example 5-33 SubmitQueueEntry Command with “file” Scheme	334
Example 5-34 SubmitQueueEntry Command with “http” Scheme	334
Example 5-35 SubmissionMethods Response	343
Example 5-36 Queue Element.....	344
Example 5-37 Custom Query.....	353
Example 5-38 Custom Response	353
Example 5-39 Custom Query for IfraTrack	354
Example 5-40 Custom Response for IfraTrack	354
Example 6-1 DieLayoutProduction: Single Shape and Two Sheet Sizes	368
Example 6-2 DieLayoutProduction: Single Shape and Range of Sheet Sizes	368
Example 6-3 DieLayoutProduction: Two Shapes and Range of Sheet Sizes	369
Example 6-4 Automated Imposition: MarkObject.....	379
Example 6-5 Imposition Template: Layout	380
Example 6-6 Output RunList (Surface)	380
Example 6-7 Automated Imposition: Ord Values	384
Example 6-8 RIPing	394
Example 6-9 Stripping: Simple Example.....	398
Example 6-10 Stripping: Complex Example	399

Example 6-11 Stitching: Combined Process	429
Example 7-1 EnumerationSpan	446
Example 8-1 Perfect Bound (Gathering).....	527
Example 8-2 Saddle-Stitched Brochure (Collecting).....	527
Example 8-3 Pseudo Code to Generate Page Count from SignatureCell Elements	538
Example 8-4 StrippingParams: Foldout Using FaceCells	541
Example 8-5 BoxFoldingParams/BoxFoldAction	547
Example 8-6 Bundle: Boxing and Palletizing	553
Example 8-7 Color	572
Example 8-8 ColorantControl: Content-Ignorant MIS	572
Example 8-9 ColorantControl: Synchronized with Input	572
Example 8-10 ColorantControl: Synchronized with Input with Alias	573
Example 8-11 ColorantControl: with ColorantAlias/ReplacementColorantName.....	573
Example 8-12 ColorantControl: with Invalid ColorantAlias/ReplacementColorantName	574
Example 8-13 ContentList.....	605
Example 8-14 ContentList: Extended with ISBN, Author, etc.....	606
Example 8-15 CylinderLayout.....	621
Example 8-16 CylinderLayout: Double-Spread-Page Plate.....	623
Example 8-17 Barcode	691
Example 8-18 PageCondition	735
Example 8-19 Layout: DynamicField Element.....	758
Example 8-20 Invalid MarkObject.....	759
Example 8-21 MarkObject	760
Example 8-22 RunList: Simple Multi-File Unseparated RunList	761
Example 8-23 RunList: Simple Multi-File Separated RunList	761
Example 8-24 OrdExpression	762
Example 8-25 DocOrd Usage	763
Example 8-26 LayoutElementProductionParams: Page Shape	768
Example 8-27 LayoutElementProductionParams: Label Shape	769
Example 8-28 LayoutElementProductionParams: Box Shape.....	769
Example 8-29 LayoutElementProductionParams: PositionObj.....	772
Example 8-30 LayoutElementProductionParams: Preflight	775
Example 8-31 LayoutPreparationParams: JDF for Figure 8-44	791
Example 8-32 LayoutPreparationParams: JDF for Figure 8-45	792
Example 8-33 LayoutShift.....	793
Example 8-34 Media: Corrugated	818
Example 8-35 Media: Self Adhesive	819
Example 8-36 Media: Flat Plate	819
Example 8-37 Media: Flexo Sleeve	819
Example 8-38 Test with InsideBox and a BoxArgument Subelement	853
Example 8-39 PRItem.....	858
Example 8-40 PrintCondition	867
Example 8-41 ProductionPath: on Path Level:	869
Example 8-42 ProductionPath: on Part Path Level:.....	870
Example 8-43 Marks and Reordering of Content using RunList/@IgnoreContext.....	898
Example 8-44 RunList: Unstructured Single-File RunList	901
Example 8-45 RunList: Multi-File Unseparated RunList	901
Example 8-46 RunList: Multi-File Unseparated RunList with Spawning	902
Example 8-47 RunList: Spawnsed RunList	902
Example 8-48 RunList: Multi-File Separated RunList	902

Example 8-49 ShapeTemplate for Figure 8-55	912
Example 9-1 DisplayGroupPool	1014
Example 9-2 FeaturePool	1015
Example 9-3 ActionPool and TestPool	1021
Example 9-4 KnownDevices Query for a Scanner	1033
Example 9-5 KnownDevices Response for a Scanner	1034
Example 9-6 KnownDevices Query for a Scanner #2	1037
Example 9-7 KnownDevices Response for a Scanner #2	1037
Example 9-8 JDF Accepted by Previous Scanner	1041
Example 9-9 JDF Rejected by Previous Scanner	1041
Example 9-10 Test for Existence of TrappedKey	1044
Example 9-11 Test for TrappedKey Equal to "Unknown"	1044
Example 9-12 Test with BoxToBoxDifference Element	1060
Example 10-1 ColorantAlias/@RawNames	1068
Example 10-2 ComChannel for Telephone	1080
Example 10-3 ComChannel for Instant Messaging	1080
Example 10-4 MetadataMap: Setting Attributes	1106
Example 10-5 RunList/MetadataMap	1106
Example 11-1 Response with Notification Element	1132
Example 11-2 Packaging of Individual JDF/JMF files in MIME	1137
Example 11-3 CID URL Scheme	1137
Example 11-4 MIME Multipart/Related	1138
Example A-1 boolean	1143
Example A-2 CMYKColor	1144
Example A-3 date	1144
Example A-4 dateTime	1144
Example A-5 DateTimeRange	1144
Example A-6 DateTimeRangeList	1145
Example A-7 double	1145
Example A-8 DoubleList	1145
Example A-9 DoubleRange	1145
Example A-10 DoubleRangeList	1146
Example A-11 duration	1146
Example A-12 DurationRange	1146
Example A-13 DurationRangeList	1146
Example A-14 gYearMonth	1147
Example A-15 hexBinary	1147
Example A-16 ID	1147
Example A-17 IDREF	1147
Example A-18 IDREFS	1148
Example A-19 integer	1148
Example A-20 IntegerList	1148
Example A-21 IntegerRange	1148
Example A-22 IntegerRangeList	1148
Example A-23 LabColor	1149
Example A-24 language	1149
Example A-25 languages	1149
Example A-26 matrix	1149
Example A-27 NameRange	1150
Example A-28 NameRangeList	1150

Example A-29 NMTOKEN.....	1150
Example A-30 NMTOKENS	1150
Example A-31 PDFPath	1151
Example A-32 rectangle.....	1151
Example A-33 RectangleRange	1151
Example A-34 RectangleRangeList	1152
Example A-35 regExp	1152
Example A-36 shape.....	1152
Example A-37 ShapeRange.....	1152
Example A-38 ShapeRangeList	1152
Example A-39 sRGBColor	1153
Example A-40 string.....	1153
Example A-41 TransferFunction.....	1153
Example A-42 URI	1154
Example A-43 URL	1154
Example A-44 URL: UTF-8	1154
Example A-45 URL: Windows Locale 1252	1154
Example A-46 URL: Escaped Characters.....	1154
Example A-47 XPath.....	1155
Example A-48 XYPair	1155
Example A-49 XYPairRange	1155
Example A-50 XYPairRangeList	1155
Example A-51 enumeration.....	1156
Example A-52 enumerations	1156
Example B-1 JDF Nodes: xsi:type	1161
Example B-2 JDF XJDF Nodes: xsi:type (not in Default Namespace)	1162
Example B-3 JMF XJMF: xsi:type	1162
Example C-1 Milestone in JMF	1177
Example I-1 @FileTemplate and @FileFormat	1206
Example N-1 FileSpec #1	1282
Example N-2 FileSpec #2	1282
Example N-3 FileSpec #3	1282
Example N-4 FileSpec #4	1283
Example N-5 FileSpec #5	1283
Example N-6 FileSpec #6	1283
Example N-7 FileSpec #7	1283
Example N-8 FileSpec #8	1284
Example N-9 FileSpec #9	1284
Example N-10 FileSpec #10	1284
Example N-11 FileSpec #11	1285
Example N-12 FileSpec #11.1 — No Partitioning	1285
Example N-13 FileSpec #11.2 — Intermediate container Partitioned	1286
Example N-14 FileSpec #11.3 — the pdf is Partitioned	1287
Example N-15 FileSpec #11.3a — the pdf is Partitioned, Different File Layout.....	1288
Example N-16 FileSpec #11.4 — Both Partitioned	1289
Example N-17 FileSpec #12	1289
Example N-18 Intent Job Ticket	1290
Example O-1 Before Processing.....	1293
Example O-2 After Processing.....	1294
Example O-3 Product JDF XJDF	1295

Example O-4 2-Component JDF XJDF before Spawning.....	1297
Example O-5 2-Component JDF Parent after Spawning the Cover Node	1298
Example O-6 2-Component JDF Spawned Node	1299
Example O-7 2-Component JDF XJDF after Merging	1300
Example O-8 Partitioned ImageSetting Node before Spawning	1301
Example O-9 Spawns Cyan Partition of the ImageSetting Node.....	1301
Example O-10 Root Partitioned ImageSetting Node after Spawning	1302
Example O-11 Merged ImageSetting Node	1303
Example O-12 RunList.....	1304
Example O-13 KnownMessages Query	1306
Example O-14 KnownMessages Response	1306
Example O-15 Status Query	1307
Example O-16 Status Response.....	1307
Example O-17 Status Signal #1	1307
Example O-18 Status Signal #2.....	1307
Example O-19 Status Signal #3.....	1308
Example O-20 Using Position	1316
Example O-21 Multiple Bindery Signatures	1317
Example O-22 Multisection Bindery Signatures.....	1317
Example O-23 Multiple Job Parts in One Imposition	1318
Example O-24 FoldOuts	1319
Example O-25 Multiple Web Layout	1320
Example O-26 Stripping Process.....	1321
Example O-27 DigitalDelivery: Before the Delivery	1323
Example O-28 DigitalDelivery: After the Delivery.....	1324
Example O-29 Delivery and DigitalDelivery Processes	1324
Example O-30 Full Example of Digital Delivery through Central Server.....	1327
Example O-31 Algorithm for Processing an Imposition Template	1330
Example O-32 Format of Variable Data Structured Content.....	1331
Example O-33 Page Pools.....	1332
Example O-34 Booklet Using Automated Imposition	1334
Example P-1 Query with Subscription to All Events.....	1342
Example P-2 Response for Subscription to All Events	1342
Example P-3 KnownControllers Query	1344
Example P-4 KnownControllers Response.....	1344
Example P-5 RepeatMessages Response	1346
Example P-6 Occupation Response	1351
Example P-7 Track Response	1353

CIP4 THANKS ITS PARTNER LEVEL MEMBERS



Kodak



HEIDELBERG

MULLER MARTINI

RICOH

xerox

