



XJDF Specification

2.0-Final

28 February 2018



CIP4 THANKS ITS PARTNER LEVEL MEMBERS



Kodak



HEIDELBERG



RICOH



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 © 2000–2018, 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 © 2000–2018, 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, Exchange Job Definition Format, XJDF, Exchange Job Messaging Format, XJMF, Job Definition Format, JDF, Job Messaging Format, JMF 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 non infringement. 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

Chapter 1 Introduction	1
1.1 Further Information	1
1.2 Background on XJDF	1
1.3 Design Criteria for XJDF	1
1.3.1 Simplify and reduce variations	1
1.3.2 Enable dynamic changes	2
1.3.3 Retain the semantic structures	2
1.3.4 Remove implementation specific details	3
1.3.5 Enhance Compatibility with standard XML and XML Tools	3
1.3.6 Device Capabilities	3
1.4 Use of XML	3
1.4.1 Use of XML Schema	3
1.4.2 Schema and XJDF Context	3
1.5 Conceptual Changes from JDF to XJDF	4
1.5.1 Use of Abstract Elements	4
1.5.2 Resource Partitioning	4
1.5.3 Structural Changes	4
1.5.4 Process Model Changes	4
1.5.5 Alignment of Signals and Audits	5
1.5.6 Messaging Changes	5
1.6 Conventions Used in This Specification	6
1.6.1 Document References	6
1.6.2 Text Styles	6
1.6.3 XPath Notation Used in this Specification	6
1.6.4 Specification of Cardinality	6
1.6.5 Template for Tables that Describe Elements	7
1.7 Glossary	7
1.8 Conformance	9
1.8.1 Conformance Terminology	9
1.8.2 Interoperability Conformance Specifications	10
1.9 Data Structures	10
1.9.1 Units of measurement	10
1.9.2 Counting in XJDF	10
Chapter 2 Overview	11
2.1 Introduction	11
2.2 Referencing Data	11
2.2.1 Referencing External Data	11
2.2.2 Identifying Sections of XJDF from External Sources	11
2.2.3 Identifying Sections of XJDF from within the Same XJDF	11
2.3 System Components	11

2.3.1 Workflow Component Roles	11
2.4 XJDF Workflow	12
2.4.1 Product Intent and Processes	12
2.5 Role of Messaging in XJDF	12
2.6 Coordinate Systems in XJDF	13
2.6.1 Introduction	13
2.6.2 Coordinate Systems of Resources and Processes	14
2.6.3 Coordinate System Transformations	15
2.6.4 General Rules	16
2.6.5 Homogeneous Coordinates	16
Chapter 3 Structure	19
3.1 XJDF	19
3.1.1 ICS Versions Value	20
3.1.2 XJDF for Product Intent	20
3.1.3 XJDF for Process Description and Gray Boxes	20
3.2 ProductList	21
3.3 ResourceSet	21
3.3.1 Dependent	22
3.4 AuditPool and Audit	23
3.4.1 AuditPool.	24
3.4.2 AuditCreated.	24
3.4.3 AuditNotification.	24
3.4.4 AuditProcessRun.	24
3.4.5 AuditResource	25
3.4.6 AuditStatus	26
3.5 XJDF Extensibility	27
3.5.1 Foreign Namespaces	27
3.5.2 Creating Extension ResourceSets.	27
3.5.3 Creating Extension Message Type Elements	27
3.5.4 Creating Extension Intent Elements	28
3.5.5 Extending NMTOKEN Lists	28
3.5.6 Extending Process Types	29
Chapter 4 Product Intent	31
4.1 ProductList	31
4.1.1 Product.	31
4.1.2 Product Amount	31
4.1.3 Intent	32
4.1.4 Product Intent	32
4.1.5 Representation of Product Binding	33
4.2 AssemblingIntent	33
4.2.1 AssemblyItem	34
4.2.2 BindIn	34

4.2.3 BlowIn	35
4.2.4 StickOn	35
4.3 BindingIntent	35
4.3.1 AdhesiveNote	37
4.3.2 EdgeGluing	37
4.3.3 HardcoverBinding	37
4.3.4 LooseBinding	39
4.3.5 SaddleStitching	40
4.3.6 SideStitching	41
4.3.7 SoftCoverBinding	41
4.3.8 Tabs	42
4.4 ColorIntent	42
4.4.1 SurfaceColor	43
4.5 ContentCheckIntent	43
4.5.1 PreflightItem	44
4.5.2 ProofItem	44
4.6 EmbossingIntent	44
4.6.1 EmbossingItem	45
4.7 FoldingIntent	45
4.8 HoleMakingIntent	46
4.9 LaminatingIntent	46
4.10 LayoutIntent	47
4.11 MediaIntent	48
4.12 ProductionIntent	49
4.13 ShapeCuttingIntent	50
4.13.1 ShapeCut	50
4.14 VariableIntent	51
Chapter 5 Processes	53
5.1 Process Template	53
5.2 Combining Individual Process Steps	54
5.2.1 Exchange ResourceSets in combined processes	54
5.2.2 Usage of ResourceSets that are used as both input and output	54
5.2.3 XJDF with Multiple Processes of the Same Type	55
5.3 General Processes	55
5.3.1 Approval	55
5.3.2 Delivery	55
5.3.3 ManualLabor	56
5.3.4 QualityControl	57
5.3.5 Verification	57
5.4 Prepress Processes	57
5.4.1 Bending	58
5.4.2 ColorCorrection	58

5.4.3 ColorSpaceConversion	58
5.4.4 DieDesign	59
5.4.5 DieLayoutProduction	59
5.4.6 ImageEnhancement	60
5.4.7 ImageSetting	61
5.4.8 Imposition	61
5.4.9 InkZoneCalculation	63
5.4.10 Interpreting	63
5.4.11 LayoutElementProduction	64
5.4.12 LayoutShifting	64
5.4.13 PDLCreation	65
5.4.14 Preflight	65
5.4.15 PreviewGeneration	66
5.4.16 RasterReading	67
5.4.17 Rendering	68
5.4.18 Screening	69
5.4.19 Separation	69
5.4.20 ShapeDefProduction	69
5.4.21 SheetOptimizing	70
5.4.22 Stripping	70
5.4.23 Trapping	73
5.5 Press Processes	73
5.5.1 ConventionalPrinting	73
5.5.2 DigitalPrinting	74
5.5.3 Varnishing	75
5.6 Postpress Processes	75
5.6.1 BlockPreparation	76
5.6.2 BoxFolding	76
5.6.3 BoxPacking	76
5.6.4 Bundling	77
5.6.5 CaseMaking	78
5.6.6 CasingIn	78
5.6.7 Collecting	79
5.6.8 CoverApplication	79
5.6.9 Creasing	80
5.6.10 Cutting	80
5.6.11 DieMaking	80
5.6.12 Embossing	81
5.6.13 EndSheetGluing	81
5.6.14 Feeding	82
5.6.15 Folding	83
5.6.16 Gathering	83
5.6.17 Gluing	84

5.6.18 HeadBandApplication	84
5.6.19 HoleMaking	85
5.6.20 Inserting	86
5.6.21 Jacketing	86
5.6.22 Labeling	87
5.6.23 Laminating	87
5.6.24 LooseBinding	88
5.6.25 Palletizing	89
5.6.26 Perforating	90
5.6.27 ShapeCutting	90
5.6.28 Shrinking	90
5.6.29 SpinePreparation	91
5.6.30 SpineTaping	91
5.6.31 Stacking.	92
5.6.32 Stitching	94
5.6.33 Strapping	95
5.6.34 ThreadSealing	95
5.6.35 ThreadSewing.	95
5.6.36 Trimming	96
5.6.37 WebInlineFinishing	97
5.6.38 Winding.	97
5.6.39 Wrapping	98
Chapter 6 Resources	99
6.1 Resource	99
6.1.1 AmountPool	100
6.1.2 PartAmount	101
6.1.3 Part	101
6.1.4 PartWaste	104
6.2 ApprovalDetails	105
6.3 ApprovalParams	106
6.4 Assembly	106
6.4.1 AssemblySection	107
6.5 BendingParams	107
6.6 BinderySignature	108
6.6.1 SignatureCell	110
6.7 BlockPreparationParams	112
6.8 BoxFoldingParams	113
6.8.1 BoxFoldingType attribute values	113
6.8.2 BoxFoldAction	116
6.9 BoxPackingParams	117
6.10 Bundle	119
6.10.1 BundleItem	119

6.11 BundlingParams121
6.12 CaseMakingParams	122
6.13 CasingInParams	123
6.14 Color	124
6.14.1 DeviceNColor	126
6.14.2 Diecutting Data (DDES3).	126
6.15 ColorantControl	127
6.15.1 ColorantAlias	128
6.15.2 DeviceNSpace	129
6.16 ColorCorrectionParams	129
6.16.1 ColorCorrectionOp	129
6.17 ColorSpaceConversionParams	130
6.17.1 ColorSpaceConversionOp	131
6.18 Component	132
6.19 Contact	134
6.19.1 ComChannel	135
6.19.2 Company	136
6.19.3 OrganizationalUnit	136
6.19.4 Person	136
6.20 Content	137
6.20.1 BarcodeProductionParams	138
6.20.2 ContentMetadata	138
6.20.3 PositionObj	139
6.21 ConventionalPrintingParams	139
6.22 CoverApplicationParams	140
6.22.1 Score	140
6.23 CreasingParams	141
6.24 CustomerInfo	142
6.25 CuttingParams	142
6.25.1 CutBlock	144
6.26 DeliveryParams	145
6.26.1 Dropltem	146
6.27 DevelopingParams	146
6.28 Device	146
6.28.1 Icon	148
6.28.2 IconList	148
6.28.3 Module	148
6.29 DieLayout	149
6.29.1 RuleLength	149
6.29.2 Station	150
6.30 DieLayoutProductionParams	150
6.30.1 RepeatDesc	150
6.31 DigitalPrintingParams	154

6.31.1 Coordinate systems in DigitalPrinting	154
6.32 EmbossingParams	155
6.32.1 Emboss	156
6.33 EndSheetGluingParams	156
6.34 ExposedMedia	157
6.35 FeedingParams	158
6.35.1 CollatingItem	158
6.35.2 Feeder	159
6.35.3 FeederQualityParams	160
6.36 FoldingParams	160
6.37 FontPolicy	161
6.38 GluingParams	161
6.39 HeadBandApplicationParams	161
6.40 HoleMakingParams	162
6.41 ImageCompressionParams	162
6.42 ImageEnhancementParams	162
6.42.1 ImageEnhancementOp	162
6.43 ImageSetterParams	162
6.44 Ink	163
6.45 InkZoneCalculationParams	163
6.46 InkZoneProfile	164
6.47 InsertingParams	164
6.48 InterpretingParams	165
6.48.1 InterpretingDetails	166
6.48.2 PDFInterpretingParams	167
6.48.3 ReferenceXObjParams	168
6.49 JacketingParams	168
6.50 LabelingParams	169
6.51 LaminatingParams	169
6.52 Layout	170
6.52.1 CIELABMeasuringField.	173
6.52.2 ColorControlStrip	173
6.52.3 Condition	174
6.52.4 ContentObject	174
6.52.5 CutMark.	174
6.52.6 DensityMeasuringField	175
6.52.7 MarkObject	175
6.52.8 PageActivation	176
6.52.9 PageCondition	176
6.52.10 PlacedObject.	177
6.52.11 Position	179
6.52.12 RegisterMark	180
6.52.13 ScavengerArea	181

6.52.14 SheetActivation	182
6.52.15 More about Layout	182
6.53 LayoutElementProductionParams	183
6.54 LayoutShift	185
6.54.1 ShiftPoint	185
6.55 LooseBindingParams	186
6.55.1 ChannelBindingDetails	187
6.55.2 CoilBindingDetails.	188
6.55.3 CombBindingDetails.	188
6.55.4 RingBindingDetails	188
6.55.5 StripBindingDetails	189
6.56 ManualLaborParams	189
6.57 Media	189
6.57.1 TabDimensions	193
6.57.2 More about Media.	195
6.58 MiscConsumable	197
6.59 NodeInfo	199
6.60 Pallet	200
6.61 PalletizingParams	200
6.62 PDLCreationParams	200
6.62.1 AdvancedParams	201
6.62.2 FontParams.	202
6.62.3 PDFCreationDetails	202
6.62.4 PDFXParams	204
6.62.5 PSCreationDetails.	204
6.62.6 ThinPDFParams.	206
6.63 PerforatingParams	207
6.64 PreflightParams	207
6.64.1 PreflightTest	207
6.65 PreflightReport.	208
6.65.1 PreflightCheck.	208
6.66 Preview	209
6.67 PreviewGenerationParams	209
6.68 QualityControlParams.	210
6.68.1 BindingQualityParams.	210
6.69 QualityControlResult	211
6.69.1 BindingQualityMeasurement	211
6.70 RasterReadingParams	211
6.71 RenderingParams	212
6.71.1 TIFFEmbeddedFile	213
6.71.2 TIFFFormatParams	213
6.71.3 TIFFtag	214
6.72 RunList	214

6.72.1 Referencing pages of a RunList from a Layout	215
6.72.2 Filtering parts of a RunList	215
6.72.3 Pages, Documents and Sets for common PDL types	217
6.72.4 Band	217
6.72.5 ByteMap	217
6.73 ScreeningParams	218
6.74 SeparationControlParams	219
6.75 ShapeCuttingParams	219
6.76 ShapeDef	219
6.77 ShapeDefProductionParams	220
6.77.1 ObjectModel	221
6.77.2 ShapeDimension	221
6.77.3 ShapeTemplate	221
6.78 SheetOptimizingParams	223
6.78.1 GangElement	223
6.79 ShrinkingParams	225
6.80 SpinePreparationParams	225
6.81 SpineTapingParams	226
6.82 StackingParams	227
6.82.1 Disjointing	228
6.82.2 InsertSheet	229
6.83 StitchingParams	230
6.84 StrappingParams	232
6.85 ThreadSealingParams	233
6.86 ThreadSewingParams	234
6.87 Tool	235
6.88 TransferCurve	236
6.89 TrappingParams	236
6.90 TrimmingParams	237
6.91 UsageCounter	237
6.92 VarnishingParams	238
6.93 VerificationParams	239
6.94 VerificationResult	239
6.95 WebInlineFinishingParams	239
6.95.1 FolderProduction	240
6.95.2 ProductionPath	240
6.96 WindingParams	240
6.97 WrappingParams	240
Chapter 7 Messaging	243
7.1 XJMF	243
7.1.1 Message	244
7.1.2 Header	244

7.2 XJMF Message Families	244
7.2.1 Query.	244
7.2.2 Command	245
7.2.3 Signal	245
7.2.4 Response	246
7.3 List of All XJMF Messages	246
7.4 ForceGang	247
7.4.1 CommandForceGang	247
7.4.2 ResponseForceGang	248
7.5 GangStatus	248
7.5.1 QueryGangStatus	248
7.5.2 ResponseGangStatus	248
7.5.3 SignalGangStatus	249
7.6 KnownDevices	249
7.6.1 QueryKnownDevices	250
7.6.2 ResponseKnownDevices	251
7.6.3 SignalKnownDevices	251
7.7 KnownMessages	251
7.7.1 QueryKnownMessages	251
7.7.2 ResponseKnownMessages	251
7.8 KnownSubscriptions	252
7.8.1 QueryKnownSubscriptions	253
7.8.2 ResponseKnownSubscriptions	253
7.8.3 SignalKnownSubscriptions	253
7.9 ModifyQueueEntry	253
7.9.1 CommandModifyQueueEntry	254
7.9.2 ResponseModifyQueueEntry	255
7.10 Notification	256
7.10.1 QueryNotification	256
7.10.2 ResponseNotification	256
7.10.3 SignalNotification	257
7.11 PipeControl	257
7.11.1 CommandPipeControl	257
7.11.2 ResponsePipeControl	258
7.12 QueueStatus	258
7.12.1 QueryQueueStatus.	258
7.12.2 ResponseQueueStatus.	258
7.12.3 SignalQueueStatus	259
7.12.4 Queue	259
7.13 RequestQueueEntry	259
7.13.1 CommandRequestQueueEntry	260
7.13.2 ResponseRequestQueueEntry	260
7.14 Resource	260

7.14.1 QueryResource	260
7.14.2 CommandResource	262
7.14.3 ResponseResource	263
7.14.4 SignalResource	265
7.15 ResubmitQueueEntry	265
7.15.1 CommandResubmitQueueEntry	266
7.15.2 ResponseResubmitQueueEntry	266
7.16 ReturnQueueEntry	267
7.16.1 CommandReturnQueueEntry	267
7.16.2 ResponseReturnQueueEntry	267
7.17 ShutDown	267
7.17.1 CommandShutDown	268
7.17.2 ResponseShutDown	268
7.18 Status	268
7.18.1 QueryStatus	268
7.18.2 ResponseStatus	269
7.18.3 JobPhase	270
7.18.4 SignalStatus	271
7.19 StopPersistentChannel	272
7.19.1 CommandStopPersistentChannel	272
7.19.2 ResponseStopPersistentChannel	273
7.20 SubmitQueueEntry	273
7.20.1 CommandSubmitQueueEntry	273
7.20.2 ResponseSubmitQueueEntry	274
7.21 WakeUp	274
7.21.1 CommandWakeUp	275
7.21.2 ResponseWakeUp	275
Chapter 8 Subelements	277
8.1 Address	277
8.1.1 AddressLine	277
8.2 ApprovalPerson	278
8.3 AutomatedOverPrintParams	278
8.4 BarcodeCompParams	278
8.5 BarcodeReproParams	279
8.6 Certification	279
8.7 Comment	280
8.8 ConvertingConfig	281
8.9 Crease	282
8.10 Cut	282
8.11 FileSpec	282
8.11.1 Disposition.	284
8.12 FitPolicy	285

8.13 Fold	285
8.14 GangSource	286
8.15 GeneralID	286
8.16 Glue	286
8.17 HolePattern	287
8.18 IdentificationField	288
8.18.1 BarcodeDetails	291
8.18.2 ExtraValues	292
8.18.3 Usage of barcode attributes	292
8.19 ImageCompression	294
8.19.1 CCITTFaxParams.	296
8.19.2 DCTParams	296
8.19.3 FlateParams	297
8.19.4 JBIG2Params	297
8.19.5 JPEG2000Params	297
8.19.6 LZWParams	298
8.20 MediaLayers	298
8.21 MetadataMap	298
8.21.1 Expr	299
8.22 MISDetails	300
8.23 Notification	301
8.23.1 Event	302
8.23.2 Milestone.	302
8.24 ObjectResolution	302
8.25 OCGControl	303
8.26 Perforate	304
8.27 QueueEntry	304
8.28 QueueFilter	305
8.29 RefAnchor	306
8.30 RegisterRibbon	306
8.31 ScreenSelector	307
8.32 Shape	308
8.33 StripMark	309
8.33.1 FillMark.	311
8.33.2 MarkColor	311
8.33.3 JobField.	311
8.34 SubscriptionInfo	312
Chapter 9 Building a System	313
9.1 Queue Support	313
9.1.1 Queue Entry ID Generation	313
9.2 Status Transitions	313
9.3 Execution Model	314

9.3.1 Determining Executable XJDF	314
9.3.2 Serial Processing	314
9.3.3 Partial Processing of XJDF with Partitioned ResourceSets.	314
9.3.4 Parallel Processing	315
9.3.5 Overlapping Processing	315
9.3.6 Approval, Proofing, Quality Control and Verification	316
9.3.7 Gang Jobs	316
9.3.8 Error Handling	317
9.4 Specifying Complex Processing	317
9.4.1 Referencing Multiple XJDF in a Directory	317
9.5 XJDF and XJMF Interchange Protocol	317
9.5.1 HTTP Port	317
9.5.2 HTTP Request Method	317
9.5.3 HTTPS-Based Protocol – TLS	318
9.6 XJMF Handshaking	318
9.6.1 Single Query/Command Response Communication	318
9.6.2 Subscribing for Signals	319
9.6.3 Managing Persistent Channels	320
9.6.4 Signal Handshaking	320
9.6.5 Reliable Signaling	320
9.6.6 Deleting Persistent Channels	321
9.6.7 XJMF Bootstrapping	321
9.6.8 Device / Controller Selection	321
9.7 XJDF Packaging	322
9.7.1 MIME Types and File Extensions	322
9.7.2 ZIP Packaging	322
9.8 Job Modification	322
9.8.1 Rescheduling with ModifyQueueEntry	322
9.8.2 Modifying Jobs	323
9.8.3 Examples for Job Modification	324
9.9 Use of XML Schema for Capability Descriptions	327
Appendix A Data Types and Values	329
A.1 XJDF Data Types	329
A.1.1 TransferFunction	331
A.2 Enumerations	331
A.2.1 Action	331
A.2.2 Activation	331
A.2.3 Anchor	332
A.2.4 Automation	333
A.2.5 Axis	333
A.2.6 BinderMaterial	333
A.2.7 BindingType	334

A.2.8 BundleType	335
A.2.9 ChannelMode	335
A.2.10 Coating	335
A.2.11 Compensation	336
A.2.12 DataType	336
A.2.13 DeviceStatus	336
A.2.14 Drying	337
A.2.15 Edge	337
A.2.16 EmbossDirection	337
A.2.17 EmbossType	337
A.2.18 Face	338
A.2.19 FeedQuality	338
A.2.20 FitPolicy	338
A.2.21 GangPolicy	339
A.2.22 Glue	339
A.2.23 IncludeResources	339
A.2.24 ISOPaperSubstrate	339
A.2.25 MappingSelection	340
A.2.26 MediaDirection	340
A.2.27 MediaType	340
A.2.28 NamedColor	341
A.2.29 Opacity	341
A.2.30 Orientation	341
A.2.31 Polarity	341
A.2.32 PositionPolicy	342
A.2.33 RenderingIntent	342
A.2.34 Scope	342
A.2.35 Severity	342
A.2.36 SheetLay	343
A.2.37 Side	343
A.2.38 Sides	343
A.2.39 SourceColorSpace	343
A.2.40 SourceObjects	346
A.2.41 StapleShape	346
A.2.42 Status	347
A.2.43 TightBacking	347
A.2.44 UpdateGranularity	348
A.2.45 Usage	348
A.2.46 WorkingDirection	348
A.2.47 WorkStyle	348
A.3 Preferred String and NMTOKEN Values	349
A.3.1 Comb and Coil Shapes	349
A.3.2 Contact Types	349

A.3.3 Content Types	350
A.3.4 Delivery Methods	351
A.3.5 Device Classes	351
A.3.6 Flute Types	353
A.3.7 Fold Catalog	353
A.3.8 Ink and Varnish Coatings	356
A.3.9 Input Tray and Output Bin Names.	356
A.3.10 MediaType Details	357
A.3.11 Milestones	358
A.3.12 Module Types	359
A.3.13 Node Categories.	362
A.3.14 Pallet Types.	363
A.3.15 Printing Technologies	363
A.3.16 PrintStandard Characterization Data Sets	364
A.3.17 Product Types.	364
A.3.18 Spine Operations	366
A.3.19 Status Details	366
A.3.20 Texture.	370
A.3.21 Units	370
Appendix B Return Values.	373
Appendix C Media Weight	375
C.1 North American Media Weight	375
C.2 Japanese Media Weight	376
C.3 Paper Grade	377
Appendix D Media Size	379
D.1 Architectural Paper Sizes	379
D.2 Business Card Sizes	379
D.3 International A Paper Sizes	379
D.4 International B Paper Sizes	380
D.5 International C Envelope Sizes	381
D.6 RA and SRA Paper Sizes	381
D.7 US ANSI Paper Sizes	381
D.8 US Paper Sizes	382
Appendix E String Generation	383
Appendix F Pagination Catalog	385
F.1 How to interpret the diagrams	385
F.1.1 Legend	385
F.1.2 Meaning of a Pagination Scheme	385
F.1.3 Modifying the Pagination Schemes with BindingOrientation	386
F.1.4 Examples of applying BindingOrientation	386

F.2 Pagination Diagrams	388
Appendix G Hole Pattern Catalog	401
G.1 Naming Scheme	401
G.2 Ring Binding - Two Hole	402
G.3 Ring Binding - Three Hole	402
G.4 Ring Binding - Four Hole	403
G.5 RingBinding - Five Hole	404
G.6 Ring Binding - Six Hole	404
G.7 Ring Binding - Seven Hole	405
G.8 Ring Binding - Eleven Hole	406
G.9 Plastic Comb Binding	406
G.10 Wire Comb Binding	407
G.11 Coil and Spiral Binding	407
G.12 Special Binding	408
Appendix H References	409

1 Introduction

This document defines the technical specification for the Exchange Job Definition Format (**XJDF**) and its counterpart, the Exchange 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 name for the XML root element: **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.

Note: The specification uses two forms for references to **XJDF/XJMF** (the general concept of the specification) and **XJDF/XJMF** (for specific reference to the root element of an XML instance).

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 act upon the data. In this first chapter, we present the concept of **XJDF**, and its relationship to **JDF** and other industry standards.

1.1 Further Information

Additional information such as application notes and examples can be found on the CIP4 website at <http://www.cip4.org>.

1.2 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 Exchange 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>).

1.3 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 modeled 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 the latest XML tools to simplify development.
- 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 **JDF** 1.x aware middleware.

1.3.1 Simplify and reduce variations

JDF 1.x allowed shorthand 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.

INTRODUCTION

1.3.1.1 Reduce the barrier of entry

Simple tasks should be easy to describe. In such cases the **XJDF** should be capable of being described as a short list of simple XPath's.

1.3.1.2 Single XJDF

JDF 1.x allowed for multiple '**JDF**' nodes within one ticket. This grouping of multiple nodes in process groups resulted in many variations of **JDF** for the same or similar requirements. Version 2.x has exactly one **XJDF** element, namely the root element; this contains no **XJDF** child nodes. This means there can be no ambiguity about where to locate and retrieve a given trait.

1.3.1.3 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.

1.3.1.4 Remove ResourceLinks

XJDF allowed specification of interdependencies of processes using '**ResourceLink**' elements. In most cases, this feature is not required if the controller maintains an internal job model. Therefore **XJDF** does not provide mechanisms to describe process networks within a single **XJDF**.

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 '**ResourceLink**' elements have been merged with their respective resources. Thus data that belongs together is also stored in the same region of the XML.

1.3.1.5 Remove RefElements

RefElements have been replaced with one of IDREF, IDREFS or inline element.

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.

1.3.1.6 Product Description

Product descriptions are now elements in their own right rather than a different type of '**JDF**' element. Thus a modification of the underlying product structure no longer modifies the overall structure of the **XJDF**. This also allows description of gang jobs where production relates to multiple products.

Intent traits are now simple attributes rather than structured spans of ranges that allow negotiation between customer and print provider. This specification assumes that any negotiation between print provider and customer takes place dynamically out of scope of this specification.

Intents that were essentially 1 to 1 copies of the respective process resource such as **DeliveryIntent** or **PackingIntent** have been removed. If the data that was provided in these intents is required for a product, then the respective process resource, e.g. **DeliveryParams** should be provided.

1.3.1.7 Imposition

JDF had three methods to describe imposed sheets: **LayoutPreparationParams** for digital printing, **StrippingParams** for MIS level imposition and **Layout** for low level RIP imposition. **XJDF** has removed **StrippingParams** and merged its properties with **Layout** which can now describe both MIS level descriptions and RIP level descriptions. Since digital printing is also moving to larger sheet sizes, **LayoutPreparationParams** have been replaced with automated **Layout**.

1.3.2 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 modeled 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.

1.3.2.1 Remove schema defaults.

All schema defaults have been removed.

1.3.3 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 straightforward. All deprecated traits have been removed completely.

1.3.4 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.

1.3.4.1 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.

1.3.5 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.

1.3.5.1 Order of Child Elements

JDF 1.x allowed for arbitrary ordering of sibling elements. This is convenient for the writer, but degrades the quality of XML schema validation because cardinality cannot be correctly enforced for unordered elements. Therefore **XJDF** generally requires sibling elements to be provided in the order as specified in the element definitions. In general the order of elements is lexically sorted in ascending order. Exceptions to alphabetical sorting will be explicitly called out in the relevant sections.

Note: Although XML is case sensitive, the ordering of elements will be determined ignoring the case of any capital letters.

Note: Attributes NEED NOT be sorted within an element.

1.3.5.2 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 resource sets can reduce data redundancy, 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.

1.3.5.2.1 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.

1.3.6 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.

1.4 Use of XML

XJDF is encoded as XML and SHALL be a valid XML document according to ▶ [XML].

Note: Most data in **XJDF** is encoded in XML attributes; XML elements provide the hierarchical structure of the data.

Note: The data model does not require use of XML. Conceptually, any hierarchical data syntax could be used. XML was chosen because it is in widespread use and in addition, leaving the choice of an underlying grammar open would lead to non interoperable implementations.

1.4.1 Use of XML Schema

The XML schema for **XJDF** is designed to ensure that **XJDF** documents are syntactically valid, thus **XJDF** documents that are successfully validated against the **XJDF** schema SHALL be considered conformant to the syntax requirements described in this specification.

1.4.2 Schema and XJDF Context.

CIP4 anticipates the uses of **XJDF** in three broad contexts:

- Original job instruction
- Change order
- Device capabilities

For original job instructions, this specification defines mandatory content that SHALL be present in the **XJDF** document. As change orders can only be used to alter an existing job, mandatory content will have been delivered to the executing

device by the original job instruction, and the change order does not need to convey this same data again. In fact, the **XJDF** document being used for a change order SHOULD only describe those values that have changed.

Note: Sending only modified values very much simplifies the executing device's task of identifying and implementing the required changes.

CIP4 provides two XML schema definitions for use with **XJDF** depending upon which context the **XJDF** document is being used in. The schema for original job instruction validates an XML document ensuring all cardinality requirements are met and can be considered to be a more rigid implementation. For change orders, most attributes and elements have been made optional in the schema which thus allows XML documents with minimum structure to be used to convey simple alterations to the consuming device. Both schemas are defined for the **XJDF** namespace: http://www.CIP4.org/JDFSchema_2_0. The namespace prefix for elements that are defined in the **XJDF** namespace should be one of 'xjdf' or no prefix, i.e. the default namespace.

For convenience the latest schema implementation can be found online as shown in the following table. Conforming XML documents NEED NOT use this in an `xsi:schemaLocation` attribute.

Table 1.1: CIP4 XJDF/XJMF Schema Locations

SCHEMA USAGE	LOCATION
XJDF job submission	http://www.CIP4.org/JDFSchema_2_0
XJDF change order	http://www.CIP4.org/JDFSchema_2_0/ChangeOrder
XJDF Device capabilities	Device specific schemas MAY be provided by the respective device vendors.

1.5 Conceptual Changes from JDF to XJDF

This section details significant structural and conceptual changes between **JDF** and **XJDF**

1.5.1 Use of Abstract Elements

The concept of abstract element types has been largely replaced by explicit element definitions. Specific details are provided in the relevant subsections in ▶ Chapter 3 Structure and ▶ Chapter 7 Messaging.

1.5.2 Resource Partitioning

Resource partitioning has been completely revised. Inheritance of abstract resource elements has been replaced by lists of resource elements within a `ResourceSet`.

1.5.3 Structural Changes

XJDF is no longer nested, there is exactly one **XJDF** element in an **XJDF** ticket. Multiple **XJDF** each with a different `@JobPartId` MAY be sent to a controller to specify multiple individual tasks.

In **JDF** terms, an **XJDF** is a Gray Box that is to be processed by a device. There are no Gray Box expansion requirements to allow a Gray Box to be 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 resource as specified in ▶ Chapter 4 Product Intent or ▶ Chapter 6 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. Product descriptions are now specified as a `ProductList` subelement of the **XJDF**. This allows informative specification of one or more products for any process, without requiring the process to be a descendent of the respective product.

1.5.4 Process Model Changes

The concept of test running has been removed. Section ▶ Section 7.6 KnownDevices SHOULD be used to query the abilities of a device. Capabilities are described as XML schema, see ▶ Section 9.9 Use of XML Schema for Capability Descriptions.

Whereas **JDF** 1.x supported explicit encoding of process networks, **XJDF** assumes that the network is implemented in a proprietary fashion by the controller. Each individual **XJDF** therefore pertains only to the receiving device. Nonetheless, this version of **XJDF** allows for a number of execution models as detailed in ▶ Section 9.3 Execution Model.

1.5.5 Alignment of Signals and Audits

In **JDF**, audits and signals were conceptually paired but syntactically slightly different. **XJDF** aligns the signals that are relevant for job costing with their respective audits. The data is syntactically identical whether it is contained in an audit or a signal, e.g. **AuditStatus** and **SignalStatus**.

1.5.6 Messaging Changes

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**.

1.5.6.1 Removal of Redundant Message Families

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 a 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: 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.

1.5.6.2 Type Safe Message Elements

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

Example 1.1: Message Type vs Explicit Message Element

A **JMF** known devices query:

```
<?xml version="1.0" encoding="UTF-8"?>
<JMF AgentName="CIP4 JDF Writer Java" AgentVersion="1.5 BLD 93"
  MaxVersion="1.6" SenderID="SenderID"
  TimeStamp="2017-05-06T16:49:46+02:00" Version="1.6"
  xmlns="http://www.CIP4.org/JDFSchema_1_1"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="JMFRootMessage">
  <!--Generated by the CIP4 Java open source JDF Library version : CIP4 JDF Writer Java 1.5 BLD 93-->
  <Query ID="m.1831._170506_164946177_000002" Type="KnownDevices" xsi:type="QueryKnownDevices">
    <DeviceFilter DeviceDetails="Brief"/>
  </Query>
</JMF>
```

Is now encoded in **XJMF** as:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--XJDF converter version: using: null null-->
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header AgentName="a1" AgentVersion="a1" DeviceID="SenderID"
    ID="1_001004" Time="2018-02-28T16:02:48+00:00"/>
  <QueryKnownDevices>
    <Header DeviceID="dummy" ID="m.1831._000002" Time="2018-02-28T16:02:48+00:00"/>
    <DeviceFilter DeviceDetails="Brief"/>
  </QueryKnownDevices>
</XJMF>
```

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

1.5.6.3 Combining Queue and Pipe Control Messages

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

All pipe control messages have been combined into a single **PipeControl** message.

1.6 Conventions Used in This Specification

This section contains conventions and notations used within this document.

1.6.1 Document References

Throughout this specification, references to other documents are indicated by short symbolic names inside square brackets (e.g., ▶ [ICC.1]). ▶ Appendix H References lists all such references, with their full title, date, source and availability.

1.6.2 Text Styles

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

- Elements are written in *blue san serif bold italic*. Examples are **Comment**, **BundleItem** and all **Resources**. This style is also used when referring to the XML **XJDF** or **XJMF** root elements.
- Attributes are written in *dark blue san serif italic*. Examples are **@Status**, **@ResourceID** and **@ID**.
- Process types are written in *purple san serif bold*. Examples are **ColorSpaceConversion** and **Rendering**.
- Enumerative and boolean values of attributes are written in *red san serif*. Examples are **"true"**, **"Waiting"**, **"Completed"** and **"Stopped"**.
- Standard **bold text** is used to highlight glossary items, defined items inside a table and definitions of local terms. Examples are **XJDF** and **XJMF** which are used when discussing either the specification in general or an instance document.
- Internal cross-reference links are denoted by a red triangle followed by gray text. An example is 'see ▶ Section 1.6.2 Text Styles'.
- External hyperlinks are denoted by blue underlined text. An example is <http://www.cip4.org>.

1.6.3 XPath Notation Used in this Specification

A simple subset of the XPath Language ▶ [XPath] is used throughout this specification to describe 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 @ symbol, e.g. "**Element/@Attribute**".
- Paths beginning with a single slash: "/" indicate root elements, e.g. **/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 an attribute="value" expression describe an element that contains an attribute with the specified value. For instance, E[**@A="V"**] specifies an element **E** whose attribute **A** has the value **V**.

1.6.4 Specification of Cardinality

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

The cardinality for **XJDF** and any child elements applies to original job instruction **XJDF** documents that are submitted to a device. In case of change orders, i.e. **XJDF** that is referenced by a **CommandResubmitQueueEntry**, the cardinality restrictions are loosened and all elements and attributes that are not required to identify the context of the change order become optional.

Note: The XML schema for change orders is designed to reflect this loosened state.

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 5.1 Process Template and ▶ Section 1.6.5 Template for Tables that Describe Elements.

Table 1.2: Cardinality Symbols (Sheet 1 of 2)

NOTATION	DESCRIPTION
T	T SHALL occur exactly once and represents an attribute or element.
T ?	T MAY occur zero or once, and represents an attribute or element. The description field MAY explain some circumstances that if met SHALL result in T occurring exactly once.
T +	T occurs one or more times, and represents an element.

Table 1.2: Cardinality Symbols (Sheet 2 of 2)

NOTATION	DESCRIPTION
T*	T occurs zero or more times, and represents an element.

1.6.5 Template for Tables that Describe Elements

Elements are defined by their attributes and sub-elements.

The ordering of the elements in the tables defines the order in which the elements SHALL appear in the respective elements.

Table 1.3: Template for Element Descriptions

NAME	DATA TYPE	DESCRIPTION
<i>Attribute-Name</i> Cardinality	Attribute- data-type	Information about the attribute.
<i>Element-Name</i> Cardinality	element	Information about the element.
<i>FileSpec</i> (<i>ResourceUsage</i>) Cardinality	element	Information about the <i>FileSpec</i> element. If " <i>ResourceUsage</i> " is specified, <i>FileSpec</i> / <i>@ResourceUsage</i> SHALL match the value specified in the parentheses. Note: When an element potentially contains multiple <i>FileSpec</i> children, the value of <i>FileSpec</i> / <i>@ResourceUsage</i> is used to distinguish them.

1.7 Glossary

The following terms are defined as they are used throughout this specification. For more detail on job and workflow components, see ▶ Section 2.3 System Components.

Table 1.4: Glossary (Sheet 1 of 3)

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 a the reference says they are.
Attribute	An XML syntactic construct describing an unstructured characteristic of an Element . See ▶ [XML] for details.
Binding Side	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> . The binding side of a signature is defined as the last fold.
Controller	The component of an XJDF based workflow that initiates Devices , routes XJDF , and communicates status information. A MIS is an example of a top level controller.
Deprecated	Indicates that an XJDF feature is being phased out of XJDF . Controllers and devices SHOULD NOT write XJDF that contains deprecated features.
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 2.3.1.2 Device 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.

Table 1.4: Glossary (Sheet 2 of 3)

TERM	DEFINITION
Face Side	The opening side of the (partial) product. This edge is opposite to the binding side.
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.
Gray Box	A Gray Box is an incomplete specification of a process. ResourceSets that are required for actual production MAY but NEED NOT be complete. Gray Boxes are typically specified by an MIS.
Input Resource	A resource is an input to a Process .
Instance Document	A document record that is part of the output of a variable data job. 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 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.
Machine	The part of a device that does not know XJDF and is controlled by an XJDF Device in a proprietary manner.
Message Family	A Message Family is a set of messages. The 4 Message Families are Query, Command, Response and Signal.
MIS	Management Information System. The functional part of an 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.
NamedFeature	The term NamedFeature describes a value that is identified by a name using XJDF/GeneralID[@DataType = "NamedFeature"] . The value is specified by GeneralID/@IDValue , and the name is specified by GeneralID/@IDUsage . For example, the GeneralID with @IDUsage="a1" and @IDValue="a2 b2" and @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: @IDValue has a type of string and therefore whitespace is allowed in the value.
Output Resource	A resource that is an output from a Process .
Partition	Resource elements within ResourceSets are partitions.
Partition Key	A Partition Key is an Attribute in Resource/Part that can identify a specific Resource within its parent ResourceSet .
PDL	Page Description Language. A generic term for any language that describes pages that 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 4 Product Intent .
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 XJMF process in a Queue . A Queue Entry MAY be comprised of multiple Workstep processes.

Table 1.4: Glossary (Sheet 3 of 3)

TERM	DEFINITION
Reader Page	A logical page as perceived by a reader. 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 .
Receiver	Device or controller that responds to an XJMF request.
Registered Side	A side on which a collection of sheets or partial products is aligned during a production step. All production steps require two registered sides, which SHALL NOT be opposite to each other. The two registered edges define the origin of the coordinate system used within the production step. When there is a binding edge, this is one of the registered sides.
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.
Sender	Device or controller that initiates an XJMF exchange.
Subelement	A child Element of some other Element .
Unique	The word “Unique” without further scope details means “Unique within the job, message or file depending on the context”. Possible scopes are: Unique within the machine (e.g., <i>ProductionPath/@ProductionPathID</i>). Unique within the workflow – This covers the whole scope at one installation (e.g., <i>StatusQuParams/@JobID</i> , <i>FileSpec/@UID</i> , <i>Device/@DeviceID</i>). 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., <i>Contact[(Part/@ContactType, "Approver")]/@ExternalID</i>).
Values include	The phrase “ Values include: ” precedes an open list of values for an Attribute whose values are of type NMTOKEN, NMTOKENS or string. The list includes recommended values but does not 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. This may be an XPath to a value defined elsewhere in the specification, or it may be a reference to a table of possible values. In either case the reference values do not include potential vendor or customer extensions.
Workstep	A workstep is an individual XJMF process that can be processed on a single device in one pass. A workstep is comprised of one or multiple Phases such as setup, production or cleanup.

1.8 Conformance

1.8.1 Conformance Terminology

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

Table 1.5: 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 1.5: Conformance Terminology (Sheet 2 of 2)

TERM	MEANING
MAY or NEED NOT	Means that an XJDF feature is truly optional.

1.8.2 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 cannot 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 is 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>.

1.9 Data Structures

Unless stated otherwise, this specification uses XML data types as defined by ▶ [XMLSchema]. For more details on **XJDF** data types, see ▶ Appendix A Data Types and Values.

1.9.1 Units of measurement

XJDF specifies most values in default units. This means that an implementation SHALL use the defined default units and SHALL NOT use alternate units.

The supported default units are described in ▶ Table A.3.21 Units which associates measurement types with the default unit. If there is no suitable entry, i.e. when a new resource is defined that introduces a new measurement type not listed in ▶ Table A.3.21 Units, then the processor MAY introduce a new unit, and that unit SHALL be based upon metric units. Speed shall be specified in units (as defined in the previous paragraph) per hour.

1.9.2 Counting in XJDF

When accessing data using an index, zero-based indices SHALL be used in **XJDF**. Thus the first index is 0, the second index is 1, etc. Negative values SHALL specify a number that is counted from the back of the list. Thus the last item is at index -1, the second to last item is at index -2 etc.

XJDF also allows ranges of items to be sub-selected from lists by using a pair of integer values where the first item identifies the start of the selection and the second item identifies the end of the selection. Thus the range "0 -1" represents all entries of a list and the range "-1 0" represents the same list in reverse order.

2 Overview

2.1 Introduction

This chapter explains the basic aspects of **XJDF**. It outlines the terminology that is used 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.

The reader is assumed to have basic knowledge of XML syntax, i.e. ▶ [XML], ▶ [XPath], ▶ [XMLNS] and ▶ [XMLSchema].

2.2 Referencing Data

2.2.1 Referencing External Data

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

2.2.2 Identifying Sections of XJDF from External Sources

Certain data elements need to be identified from multiple **XJDF** or **XJMF** instances. Examples include **XJDF**/[@JobID](#), **XJDF**/[@JobPartID](#), [@ExternalID](#) and all attributes of **Part**. These entities do NOT have a data type of ID and systems that use **XJDF** SHALL maintain them.

An individual workstep SHALL be uniquely identified by the combination of [@JobID](#), [@JobPartID](#) and the partition keys defined in any **Part**.

2.2.3 Identifying Sections of XJDF from within the Same XJDF

Certain data elements 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.

The [@ID](#) attribute is generally defined in generic elements such as **Header**, **Product** or **Resource**. When the referencing element requires to reference a specific element such as **Media**, the [@ID](#) attribute will be that of the containing generic element.

Example: An attribute [@MediaRef](#) will reference **Resource**/[@ID](#) of the respective parent of the **Media**.

2.3 System Components

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

2.3.1 Workflow Component Roles

The components that create, modify, route, interpret and execute an **XJDF** job are known as controllers, queues, 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. In practice, it is very likely that individual system components will include a mixture of the roles described in the following sections.

2.3.1.1 Machine

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.

2.3.1.2 Device

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 ex-

ecution. The communication between machines and devices is by definition proprietary and therefore not defined in this specification. Devices SHOULD support **XJMF** messaging in order to interact dynamically with a ▶ Controller.

2.3.1.3 Queue

Whereas a ▶ Device processes **XJDF** to produce a result, queues provide a method of ordering, prioritizing and scheduling queue entries 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 queue entry and cannot store any waiting queue entries still implements an albeit minimalistic queue.

2.3.1.4 Controller

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 controller. Controllers can also determine ▶ Process planning and scheduling data, such as ▶ Process times and planned production amounts.

2.3.1.5 Management Information System—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**.

2.4 XJDF Workflow

XJDF does not dictate that a workflow must be constructed in any particular 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 prepress 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, raster image processing, 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.

2.4.1 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 4 Product Intent.
- **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 6 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.

2.5 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 7 Messaging and ▶ Chapter 9 Building a System.

2.6 Coordinate Systems in XJDF

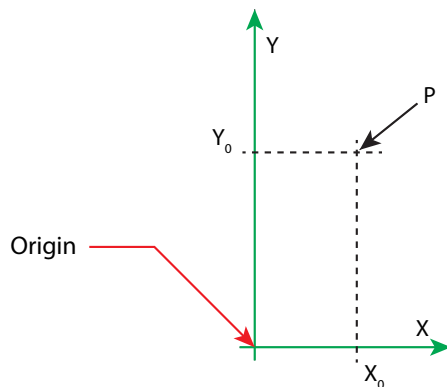
This chapter explains how coordinate systems are defined and used in **XJDF**. It also shows how the matrices are used to specify a certain transformation and how these matrices can be used to transform coordinates from one coordinate system to another coordinate system.

2.6.1 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 that is then 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 2-1: Standard coordinate system

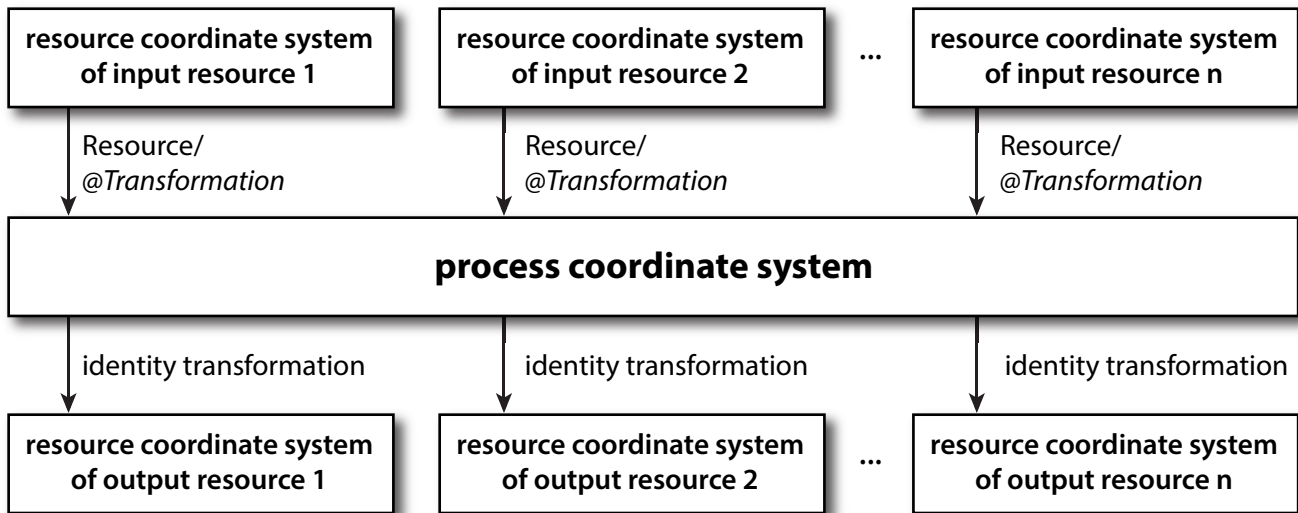


Each page contained in a PDL file has its own coordinate system. In the same way a piece of film or a sheet of paper has a coordinate system. Within **XJDF** each of these coordinate systems is called a *resource coordinate system*.

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 2.6.5 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 2-2: 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 [Resource/@Orientation](#) or [Resource/@Transformation](#).

2.6.1.1 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.

2.6.2 Coordinate Systems of Resources and Processes

Each physical resource (e.g., [Component](#)) of a process has 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 [Resource/@Orientation](#) or [Resource/@Transformation](#). This provides the ability to specify different matrices for the individual resources of a process.

2.6.2.1 Use of Preview to Display Resource Orientation

It is often necessary to load printed material into finishing equipment manually. Particularly in the case of imposed sheets, the page orientation will not be unique and even the concept of "Front" or "Back" can be confusing, since front and back pages can be printed on the same surface of the imposed sheet. [Preview ResourceSets](#) with [Part/@PreviewType](#) = "ThumbNail" or [Part/@PreviewType](#) = "Viewable" SHOULD be provided to illustrate the desired orientation of the input components with respect to the device.

2.6.2.2 Coordinate Systems of Combined Processes

[XJDF/@Types](#) MAY specify multiple individual processes and thus also the respective coordinate systems of those processes. The individual process coordinate systems are not modified by the fact that the processes are part of a combined process. The orientation of a [Resource](#) for a specific process can be modified by specifying [Resource/@Orientation](#) or [Resource/@Transformation](#). The resources that apply to a given process are defined explicitly in the process tables in

► Chapter 5 Processes for a mapping of parameter resources to process types.

2.6.3 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 width (**w**) and the height (**h**) of the **Component** as specified by X and Y in **Component/@Dimensions**. If these are unknown, it is still possible to define a general orientation in **Resource/@Orientation**. 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 2.1: Matrices and Orientation values for describing the orientation of a Component (Sheet 1 of 2)

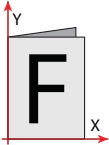
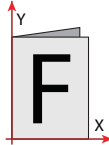
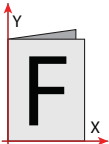
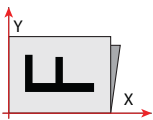
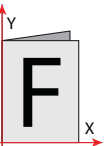
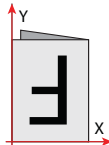
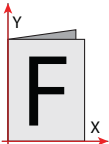
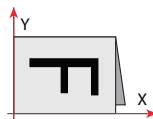
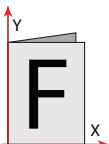
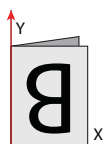
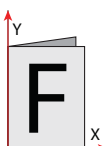
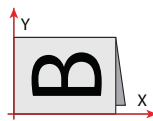
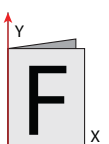
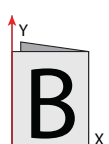

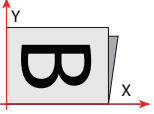
ORIENTATION VALUE	SOURCE COORDINATE SYSTEM	TRANSFORMATION MATRIX ACCORDING ACTION	TARGET COORDINATE SYSTEM
Rotate0		$1 \ 0 \ 0 \ 1 \ 0 \ 0$ No Action	
Rotate90		$0 \ 1 \ -1 \ 0 \ h \ 0$ 90° Counterclockwise Rotation	
Rotate180		$-1 \ 0 \ 0 \ -1 \ w \ h$ 180° Rotation	
Rotate270		$0 \ -1 \ 1 \ 0 \ 0 \ w$ 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 2.1: 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
Flip270		$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}$ 270° Counterclockwise Rotation + Flip around X	

2.6.4 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 the particular Machine 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.

2.6.5 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 SHALL be set to "0 0 1".

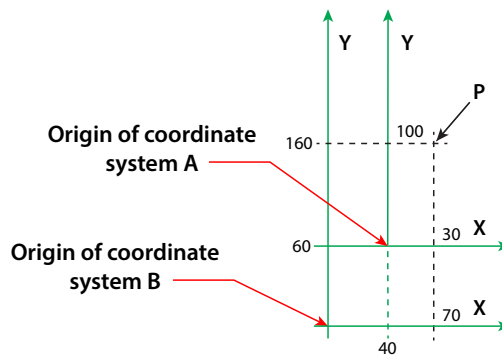
Table 2.2: Coordinate Transformation Examples

MATRIX	XJDF VALUE	DESCRIPTION
$\begin{bmatrix} a & b & 0 \\ c & d & 0 \\ e & f & 1 \end{bmatrix}$	"a b c d e f"	General transformation case.
$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	"1 0 0 1 0 0"	Identity transformation.
$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ dx & dy & 1 \end{bmatrix}$	"1 0 0 1 dx dy"	Translation by dx, dy.
$\begin{bmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$	"cosφ sinφ -sinφ cosφ 0 0"	Rotation by φ degrees counter-clockwise.

2.6.5.1 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 2-3: Transforming a point (example)



Transformation sequence

$P_A = [30 \ 100 \ 1]$	Starting position $P_A = (30, 100)$
$P_B = P_A \times Trf$	Transformation
$P_B = [30 \ 100 \ 1] \times \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 40 & 60 & 1 \end{bmatrix}$	Expanded translation transformation. In XJDF, Trf is written as an attribute with a data type of matrix, e.g. <code>@CTM="1 0 0 1 40 60"</code>
$P_B = [70 \ 160 \ 1]$	Result position $P_B = (70, 160)$

3 Structure

A single **XJDF** 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** 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 a separate **XJDF** for each work step. These MAY be packaged together and submitted as one or more transactions. See ▶ Chapter 9 Building a System for details of packaging and referencing of the individual **XJDF**.

3.1 XJDF

The top-level element of an **XJDF** instance SHALL be an **XJDF** element. See ▶ Table 3.1 XJDF 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** 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.

Table 3.1: XJDF (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Category</i> ?	NMTOKEN	@Category specifies the named category of this XJDF . Controllers SHOULD specify @Category for processes that have many optional values in @Types . This allows processors to identify the general purpose of an XJDF without parsing the @Types field. For instance, a RIP for final output and a RIP for proof process have identical @Types attribute values, but have @Category = "ProofRIPing" or @Category = "RIPing", respectively. Values include those from: ▶ Node Categories. Note: @Category MAY also be the name of a gray box defined by an ICS document. See ▶ Section 1.8.2 Interoperability Conformance Specifications for details.
<i>CommentURL</i> ?	URL	URL SHALL refer to an external, human-readable description of this XJDF .
<i>DescriptiveName</i> ?	string	Human-readable descriptive name of this XJDF . @DescriptiveName SHOULD be provided for communication from applications to humans in order to reference the XJDF .
<i>ICSVersions</i> ?	NMTOKENS	@ICSVersions SHALL list all CIP4 Interoperability Conformance Specification (ICS) Versions that this XJDF complies with. The value of @ICSVersions SHALL conform to the value format described in ▶ Section 3.1.1 ICS Versions Value.
<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	@JobPartID SHALL identify one or more worksteps of the same type that can be described as one XJDF . @JobPartID is internal to the MIS system that created the XJDF .
<i>ProjectID</i> ?	NMTOKEN	Identification of the project context that this XJDF belongs to. @ProjectID SHOULD be used by a controller to group a set of XJDF jobs.
<i>RelatedJobID</i> ?	NMTOKEN	Job identification of a related job. Used to identify the @JobID 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.

Table 3.1: XJDF (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<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>RelatedProjectID</i> ?	NMTOKEN	Identification of a related project context that this XJDF belongs to. <i>@RelatedProjectID</i> SHOULD be used by a controller to group a set of XJDF jobs.
<i>Types</i>	NMTOKENS	A list of one or more process names that are specified within this XJDF document. For details on using processes, see ▶ Section 3.1.3 XJDF for Process Description and Gray Boxes. A value of <i>@Types</i> that contains "Product" 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 "Product". Values include those from: ▶ Chapter 5 Processes.
<i>xmlns</i>	URI	XJDF supports use of XML namespaces. For details on using namespaces in XML, see ▶ [XMLNS]. For version 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. <i>AuditPool</i> elements are intended to serve the requirements of MIS for evaluation and post calculation. See ▶ Section 3.4 AuditPool and Audit.
<i>Comment</i> *	element	Any human-readable text. The <i>Comment</i> element is different from an XML comment <code><!-- XML Comment --></code> . The XJDF comment is meant for display in a user interface whereas the XML comment is used to add developer's 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.
<i>ResourceSet</i> *	element	Container elements for <i>Resource</i> elements.

3.1.1 ICS Versions Value

To assist with interoperability conformance the **XJDF** can refer to one or more **CIP4** Interoperability Conformance Specification documents. Each document is referenced by using an NMTOKEN that complies with the following:

Value format: `<ICSName>_L<ICSLevel>-<ICSVersion>`.

Example: MISPRE_L1-2.0 for the MIS to Prepress ICS.

3.1.2 XJDF for Product Intent

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

XJDF/@Types SHALL have a value of "Product" to indicate that the **XJDF** does not specify any processing.

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

3.1.3 XJDF for Process Description and Gray Boxes

Process **XJDF** contain processing instructions in *ResourceSet* elements that are targeted to a specific device in addition to an optional product definition in a *ProductList* and intent elements. *@Types* of process **XJDF** SHALL NOT contain the token "Product" if any additional process type tokens are present. In some cases such as prepress, a controller such as an MIS will not know all the details of the process including the exact list of *XJDF/@Types* or any details of the *ResourceSets* for the respective processes. It can still provide an **XJDF** with the limited, known information. These limited **XJDF** are referred to as Gray Boxes.

Note: There is no syntactical difference between a gray box and a dedicated process **XJDF**. The boundary between Gray Boxes and dedicated process **XJDF** is very fuzzy, since devices will typically apply defaults to any data that is missing in a process **XJDF**.

3.1.3.1 Specifying NamedFeatures with GeneralID

XJDF 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.

3.2 ProductList

The **ProductList** specifies a list of products and product parts from the print buyer’s point of view. For more details on product intent, see ▶ Chapter 4 Product Intent.

Table 3.2: 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.

3.3 ResourceSet

A **ResourceSet** describes a set of one or more **Resource** elements that are logically grouped together. A **ResourceSet** can describe either physical entities such as paper or logical entities such as process parameters. **ResourceSet** elements with the same values of @Name, @Usage, @ProcessUsage and common or no entries in @CombinedProcessIndex SHALL NOT be specified.

Note: This restriction is designed to ensure that the applicable **ResourceSet** for a process can be unambiguously identified.

For instance a **ResourceSet**[@Name="NodeInfo"] MAY be defined for end customer scheduling requirements by specifying **ResourceSet**/@ProcessUsage="EndCustomer" and partitioning by @ProductPart. Then the production scheduling SHOULD be defined in a separate **ResourceSet**[@Name="NodeInfo"] without @ProcessUsage.

An individual **Resource** SHALL be referenced by referencing **Resource**/@ID. The **ResourceSet** SHALL be referenced by referencing **ResourceSet**/@ID. Unless otherwise specified, an @IDREF or @IDREFS will refer to an individual **Resource** rather than an entire **ResourceSet**.

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

Table 3.3: ResourceSet Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
CombinedProcessIndex ?	IntegerList	@CombinedProcessIndex specifies the zero based indices of individual processes within the complete list of XJDF /@Types that this ResourceSet SHALL apply to. Multiple entries in @CombinedProcessIndex specify that the ResourceSet is used by the respective multiple processes. @CombinedProcessIndex SHALL be specified if multiple ResourceSet items with the same @Name, @ProcessUsage and @Usage are specified in one XJDF . If @CombinedProcessIndex is not specified, the ResourceSet applies to all processes that match the @Name, @ProcessUsage and @Usage requirements as listed in ▶ Chapter 5 Processes.
CommentURL ?	URL	URL to an external, human-readable description of the ResourceSet .
DescriptiveName ?	string	Human-readable descriptive name of the ResourceSet .
ID ?	ID	Identifier of the ResourceSet .
Name	NMTOKEN	@Name SHALL specify the name of the explicit resource that this ResourceSet represents. Child resource elements of this ResourceSet SHALL NOT contain resources that do not match @Name. @Name of resource types that are specified in ▶ Chapter 6 Resources of this specification SHALL be provided without an XML namespace prefix. @Name of proprietary resources SHALL be provided with an XML namespace prefix. See ▶ Section 3.5 XJDF Extensibility for details. A list of predefined resources is specified in ▶ Chapter 6 Resources.

Table 3.3: ResourceSet Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ProcessUsage</i> ?	NMTOKEN	@ <i>ProcessUsage</i> identifies the context of a <i>Resource</i> if multiple <i>Resource</i> elements of the same type are supplied for an individual process type. Values include those specified in the appropriate process descriptions in ▶ Chapter 5 Processes. Note: ICS documents MAY define additional values for @ <i>ProcessUsage</i> .
<i>Unit</i> ?	NMTOKEN	Unit of measurement for the values of <i>AmountPool/PartAmount/@Amount</i> , <i>AmountPool/PartAmount/@MaxAmount</i> , <i>AmountPool/PartAmount/@MinAmount</i> and <i>AmountPool/PartAmount/@Waste</i> . Values include those from: ▶ Units. Note: Units other than those defined in the above table SHOULD NOT be specified.
<i>Usage</i> ?	enumeration	@ <i>Usage</i> shows that the resource is either consumed or produced within this XJDF document. If no @ <i>Usage</i> is specified, the <i>ResourceSet</i> or its <i>Resource</i> children SHALL contain @ <i>ID</i> and be referenced from elsewhere within the XJDF . Allowed value is from: ▶ Usage. Note: <i>ResourceSet</i> [@ <i>Usage</i> ="Output"] MAY contain data that is conceptually input data for the XJDF .
<i>Comment</i> *	element	Any human-readable text that relates to the <i>ResourceSet</i> .
<i>Dependent</i> *	element	Reference to an XJDF that produces this <i>ResourceSet</i> [@ <i>Usage</i> ="Input"] or consumes this <i>ResourceSet</i> [@ <i>Usage</i> ="Output"]. Multiple <i>Dependent</i> elements specify that the <i>Dependent</i> relates to multiple consuming or producing processes.
<i>GeneralID</i> *	element	Additional identifiers related to the <i>ResourceSet</i> .
<i>Resource</i> *	element	List of <i>Resource</i> elements.

Example 3.1: ResourceSet with CombinedProcessIndex

The following example shows the use of *ResourceSet*/@*CombinedProcessIndex* to differentiate the scheduling for a finishing combined process that contains both **Cutting** and **Folding**. The *NodeInfo* with @*CombinedProcessIndex*="0" applies to the first token in *XJDF*/@*Types*, i.e. **Cutting**, whereas the *NodeInfo* with @*CombinedProcessIndex*="1" applies to the second token in *XJDF*/@*Types*, i.e. **Folding**. Since *CuttingParams* is uniquely linked to the **Cutting** process, and similarly *FoldingParams* is uniquely linked to the **Folding** process (see ▶ Cutting – Input Resources and ▶ Folding – Input Resources), @*CombinedProcessIndex* NEED NOT be specified for those resources.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="CPI_Example" Types="Cutting Folding">
  <ResourceSet CombinedProcessIndex="0" Name="NodeInfo" Usage="Input">
    <Resource>
      <NodeInfo Start="2018-02-28T13:00:00+00:00"/>
    </Resource>
  </ResourceSet>
  <ResourceSet CombinedProcessIndex="1" Name="NodeInfo" Usage="Input">
    <Resource>
      <NodeInfo Start="2018-02-28T17:00:00+00:00"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="CuttingParams" Usage="Input">
    <Resource/>
  </ResourceSet>
  <ResourceSet Name="FoldingParams" Usage="Input">
    <Resource/>
  </ResourceSet>
</XJDF>
```

3.3.1 Dependent

A *Dependent* element SHALL reference an **XJDF** that produces an input *ResourceSet* or consumes an output *ResourceSet*. The data in *Dependent* elements allows devices to communicate directly with other devices in the workflow that are pro-

cessing the same job. The data provided in **Dependent** also provides pipe control information. See ▶ Section 9.3.5 Overlapping Processing and ▶ Section 7.11 PipeControl.

Table 3.4: *Dependent Element*

NAME	DATA TYPE	DESCRIPTION
<i>JobID</i>	NMTOKEN	@ <i>JobID</i> of the referenced process. Note: @ <i>JobID</i> will typically match XJDF / <i>@JobID</i> unless parts of the job are being produced on gang forms.
<i>JobPartID</i> ?	NMTOKEN	@ <i>JobPartID</i> of the referenced process. @ <i>JobPartID</i> SHALL NOT match XJDF / <i>@JobPartID</i> .
<i>PipeID</i> ?	NMTOKEN	If this attribute exists, the resource is a pipe. @ <i>PipeID</i> is used by XJMF pipe-control messages to identify the pipe. For more information, see ▶ Section 9.3.5 Overlapping Processing.
<i>PipeProtocol</i> ?	NMTOKEN	@ <i>PipeProtocol</i> defines the protocol use for pipe handling. Proprietary pipe protocols MAY be specified in addition to those defined below but will not necessarily be inter-operable. Values include: IdentificationField – The pipe data is provided by barcodes that are defined in IdentificationField elements. XJMF – XJMF based PipeControl messages. The sequence of pipe initialization is undefined. See next two values: " XJMFPush " and " XJMFPull ". XJMFPush – XJMF based PipeControl protocol. The producing device initiates the protocol. XJMFPull – XJMF based PipeControl protocol. The consuming device initiates the protocol. None – No pipe support.
<i>XJMFURL</i> ?	URL	URL of a processor that has knowledge of the referenced process. The processor at this URL MAY be queried for additional information using XJMF .

3.4 AuditPool and Audit

AuditPool elements contain the post-facto recorded results of a process. Audits are conceptually very similar to job-specific signals. Signals record the current state of a process or device, whereas audits summarize that device state over a longer period during the execution of a single process. Thus an audit will summarize the result of multiple signals belonging to a unique phase in the returned **XJDF**. A unique phase SHOULD contain the same combination of **JobPhase**/**@Status** and **JobPhase**/**@StatusDetails**. Thus minor variations such as speed NEED NOT be recorded as separate audits although they MAY have slightly varying values in the respective signals. **AuditPool** elements record any event related to the situations described in ▶ Table 3.5 Alignment of Audits and Messages.

Note: Audits are always in the context of a process. Thus job independent signals such as the **SignalStatus** of an idle device will never be tracked as an audit.

Note: The data in **XJMF** responses are very similar to the data in **XJMF** signals. The only difference is that **XJMF** responses are synchronous HTTP responses. Therefore, all discussions referring to signals in this section apply equally to responses.

Table 3.5: *Alignment of Audits and Messages*

AUDIT	SIGNAL	COMMENT
AuditCreated	-	First audit in the XJDF .
AuditNotification	SignalNotification	One AuditNotification SHOULD be written for each SignalNotification .
AuditProcessRun	CommandReturnQueueEntry	A process run SHOULD be written whenever an XJDF is returned to the controller by a device.
AuditResource	SignalResource	One AuditResource SHOULD be written with the final data for each unique phase.
AuditStatus	SignalStatus	One AuditStatus SHOULD be written with the final data for each phase.

STRUCTURE

Audit information might be used by MIS for operations such as evaluation or invoicing. **AuditPool** entries are ordered chronologically, with the last entry in the **AuditPool** representing the newest. An **AuditProcessRun** element shall finalize each ▶ Workstep. All subsequent entries in the **AuditPool** belong to the next ▶ Workstep.

3.4.1 AuditPool

The following table defines the contents of the **AuditPool** element. In contrast to most other elements in **XJDF** the child elements of **AuditPool** SHALL be ordered chronologically from oldest to newest rather than alphabetically.

Table 3.6: AuditPool Element

NAME	DATA TYPE	DESCRIPTION
AuditCreated *	element	Logs creation of an XJDF or creation of a resource.
AuditNotification *	element	Logs individual events that occurred during processing.
AuditProcessRun *	element	Summarizes one complete execution run of an XJDF or delimits a group of AuditPool elements for each individual process run.
AuditResource *	element	Describes the usage of resources during execution of an XJDF or the modification of the intended usage of a resource.
AuditStatus *	element	Logs start and end times of any process states and sub-states, denoted as phases. Phases can reflect any arbitrary subdivisions of a process.

3.4.2 AuditCreated

This element allows the creation of an **XJDF** or resource to be logged.

Table 3.7: AuditCreated Element

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .

3.4.3 AuditNotification

AuditNotification contains information about individual events that occurred during processing. For a detailed discussion of event properties, see ▶ Section 9.3.8 Error Handling.

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

Table 3.8: AuditNotification Element

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
Notification	element	Notification that describes the event. See ▶ Section 8.23.1 Event and ▶ Section 8.23.2 Milestone.

3.4.4 AuditProcessRun

AuditProcessRun summarizes one execution of a process. An **AuditProcessRun** SHALL be written each time an **XJDF** is returned to a controller.

All job related amounts in subsequent **AuditPool** elements and **XJMF** messages SHALL restart at 0 when an **XJDF** is processed on a device after a **AuditProcessRun** has been sent.

Table 3.9: AuditProcessRun Element

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
ProcessRun	element	Details of the individual process execution.

3.4.4.1 ProcessRun

The **ProcessRun** element contains the details of the individual process execution.

Table 3.10: ProcessRun Element

NAME	DATA TYPE	DESCRIPTION
<i>Duration</i> ?	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 NodeInfo/@Status is "InProgress", "Setup" or "Cleanup".
<i>End</i>	dateTime	Date and time at which the process ended.
<i>EndStatus</i>	enumeration	The NodeInfo/@Status of the process at the end of the run. For a description of process states, see ▶ Appendix A.2.4.2 Status. Allowed values are: Aborted – The XJDF has been aborted before producing the desired result. Completed – The XJDF has been completed and the desired result has been produced.
<i>QueueEntryID</i> ?	NMTOKEN	@QueueEntryID of the QueueEntry for which this AuditProcessRun was generated.
<i>ReturnTime</i> ?	dateTime	Date and time of the ReturnQueueEntry submission.
<i>Start</i>	dateTime	Date and time at which the process started.
<i>SubmissionTime</i> ?	dateTime	Date and time of the SubmitQueueEntry submission. This value SHOULD be identical with QueueEntry/@SubmissionTime .
Part *	element	Describes which parts of a process this ProcessRun belongs to. If Part is not specified for a ProcessRun , it refers to all parts.

3.4.5 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 **AuditProcessRun** as an **SignalResource**.

Table 3.11: AuditResource Element

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See Message/Header .
<i>ResourceInfo</i>	element	ResourceInfo describes the consumption or production of an individual Resource . ResourceInfo/ResourceSet/Resource elements NEED NOT contain the explicit resources as defined in ResourceInfo/ResourceSet/@Name .

Example 3.2: AuditResource: Logging of Consumption

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

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="PaperAudit" Types="ConventionalPrinting">
  <AuditPool>
    <AuditCreated>
      <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:14+00:00"/>
    </AuditCreated>
    <AuditResource>
      <Header DeviceID="TestSender" ID="1_000007" Time="2018-02-28T16:00:14+00:00"/>
      <ResourceInfo>
        <ResourceSet Name="Component" Usage="Input">
          <Resource>
            <AmountPool>
              <PartAmount Amount="400" Waste="21"/>
            </AmountPool>
            <Part SheetName="S1"/>
          </Resource>
        </ResourceSet>
      </ResourceInfo>
    </AuditResource>
    <AuditResource>
      <Header DeviceID="TestSender" ID="1_000009" Time="2018-02-28T16:00:14+00:00"/>
      <ResourceInfo>
        <ResourceSet Name="Media">
          <Resource>
            <Media MediaType="Paper" Weight="90"/>
          </Resource>
        </ResourceSet>
      </ResourceInfo>
    </AuditResource>
  </AuditPool>
  <ResourceSet Name="Media">
    <Resource ID="Media_000004.1">
      <Part SheetName="S1"/>
      <Media MediaType="Paper" Weight="80"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Component" Usage="Input">
    <Resource>
      <AmountPool>
        <PartAmount Amount="400"/>
      </AmountPool>
      <Part SheetName="S1"/>
      <Component MediaRef="Media_000004.1"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

3.4.6 AuditStatus

AuditStatus contains audit information about the start and end times of any process states and sub-states, 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/Header/@DeviceID** of the fork lift and specifies the actual start and end time of the usage of that fork lift.

Table 3.12: AuditStatus Element

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
DeviceInfo	element	DeviceInfo describes details of the actual device status.

3.5 XJDF Extensibility

The **XJDF** specification aims to support plug-and-play as much as possible. Nonetheless, **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. **XJDF** extensibility SHALL be implemented using XML namespaces; see ▶ [XMLNS].

3.5.1 Foreign Namespaces

Attributes in a foreign namespace MAY be added to any **XJDF** element.

Elements in a foreign namespace SHALL NOT be specified in any **XJDF** element other than the **Notification**, **Resource** and **Intent** elements. The children of these elements SHALL be ordered so that all elements in a foreign namespace follow all of the elements in the **XJDF** namespace.

Example 3.3: Namespaces in XML

The example illustrates how private namespaces are declared and used to extend an existing **XJDF Media** element by adding a private attribute and a namespace declaration.

```
<Resource>
  <Part SheetName="S1"/>
  <Media MediaType="Paper" xmlns:foo="http://www.foo.org" foo:FooAtt="FooVal"/>
</Resource>
```

3.5.2 Creating Extension ResourceSets

New types of **ResourceSet** may be defined by creating a **ResourceSet** with **@Name** referring to a proprietary xml namespace. The extension element SHALL reside in the appropriate child **ResourceSet/Resource** element.

Example 3.4: Creating Extension ResourceSets

```
<ResourceSet ID="FooParams_000004" Name="foo:FooParams" Usage="Input"
  xmlns:foo="http://www.foo.org">
  <Resource ID="FooParams_000004.1">
    <Part Run="R1"/>
    <foo:FooParams FooAtt="FooVal"/>
  </Resource>
</ResourceSet>
```

3.5.3 Creating Extension Message Type Elements

New message types may be defined by creating a message in a proprietary xml namespace that adheres to the naming scheme of **XJMF**.

The extension message SHALL reside in the **XJMF** element. Extension messages SHOULD follow the naming scheme using message family and type and SHOULD contain a **Header** element. The following example shows a query and its matching response for a new message type in the “foo” namespace.

Example 3.5: Creating Extension Messages

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0" xmlns:foo="www.foo.org">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:17+00:00"/>
  <foo:QueryBar>
    <foo:BarParams BarDetails="value"/>
    <Header DeviceID="TestSender" ID="queryID" Time="2018-02-28T16:00:17+00:00"/>
  </foo:QueryBar>
</XJMF>

<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0" xmlns:foo="www.foo.org">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:17+00:00"/>
  <foo:ResponseBar>
    <foo:BarResonseParams BarDetails="value"/>
    <Header DeviceID="TestSender" ID="1_000003"
      Time="2018-02-28T16:00:17+00:00" refID="queryID"/>
  </foo:ResponseBar>
</XJMF>
```

Example 3.6: Creating Mixed Extension Messages

The following example shows how **XJMF** messages can be mixed and interleaved with extension messages.

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0" xmlns:foo="www.foo.org">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:21+00:00"/>
  <QueryKnownDevices>
    <Header DeviceID="TestSender" ID="Q1" Time="2018-02-28T16:00:21+00:00"/>
  </QueryKnownDevices>
  <foo:QueryBar>
    <Header DeviceID="TestSender" ID="F1" Time="2018-02-28T16:00:21+00:00"/>
    <foo:BarParams BarDetails="value"/>
  </foo:QueryBar>
  <QueryKnownMessages>
    <Header DeviceID="TestSender" ID="Q2" Time="2018-02-28T16:00:21+00:00"/>
  </QueryKnownMessages>
</XJMF>
```

3.5.4 Creating Extension Intent Elements

New intent elements may be defined by creating an intent with *@Name* referring to a proprietary xml namespace. The extension element SHALL reside in the intent element.

Example 3.7: Creating Extension Intent elements

```
<Product IsRoot="true">
  <Intent Name="foo:FooIntent">
    <foo:FooIntent xmlns:foo="http://www.foo.org" FooAtt="FooVal"/>
  </Intent>
</Product>
```

3.5.5 Extending NMTOKEN Lists

Some elements contain attributes of type NMTOKEN and some of these have a set of predefined suggested values. These sets are open by design and MAY be extended with other values providing such additional values do not conflict with the usage of those already defined in this specification.

Additional values MAY use a namespace like syntax (i.e., a namespace prefix separated by a single colon ":"), in which case the namespace prefix SHOULD be defined in the **XJDF** ticket with the standard xmlns:Prefix="someURI" notation, even if no other use of that namespace occurs in the **XJDF** ticket. Implementations that find an unknown NMTOKEN that has a namespace prefix MAY then attempt to use its default value of that attribute.

For other NMTOKEN lists that have a pre-defined meaning or employ a specific syntax (e.g. *@Separation* or *@FoldCatalog*), additional values SHOULD NOT use the namespace prefix format but SHOULD conform to the usage for that data type, i.e. a new value for *@Separation* SHOULD be the name of a separation employed within the **XJDF**. Similarly, a new value of *@FoldCatalog* SHOULD conform to the normal 'Fx-y' syntax. Implementations that find an unknown NMTOKEN without a namespace prefix MAY then raise an error.

3.5.6 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 3.1.3 XJDF for Process Description and Gray Boxes) can create process **XJDF** 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

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="IntentExtension"
  Types="foo:FooMaking" xmlns:foo="http://www.foo.org">
</XJDF>
```

3.5.6.1 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.

4 Product Intent

Product Intent provides a description of finished products from the print buyer's point of view.

4.1 ProductList

The products or set of products that are processed during a given workstep MAY be specified in a **ProductList**, which 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.

4.1.1 Product

The **Product** element specifies an individual product or product part.

Table 4.1: Product Element

NAME	DATA TYPE	DESCRIPTION
<i>Amount</i> ?	integer	Total number of products or product parts of this type to produce. For product parts, if present <i>@Amount</i> SHALL specify the number of copies of the product part that are needed in order to produce one product.
<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	Internal identifier of this product.
<i>IsRoot</i> ?	boolean	If true, this Product is a self-contained product. If false, this Product is a product part of another Product , such as a cover or insert. Multiple Product elements with <i>@IsRoot="true"</i> MAY be specified, for instance in a gang job. If the parent ProductList element contains multiple Product elements, <i>@IsRoot</i> SHOULD be specified.
<i>MaxAmount</i> ?	integer	Maximum total number of products to produce including the maximum overage that the customer is willing to accept. <i>@MaxAmount</i> SHOULD NOT be specified for product parts.
<i>MinAmount</i> ?	integer	Minimum total number of products to produce including the maximum underage that the customer is willing to accept. <i>@MinAmount</i> SHOULD NOT be specified for product parts.
<i>ProductType</i> ?	NMTOKEN	Classification of this product or product part. Values include those from: ▶ Product Types.
<i>ProductTypeDetails</i> ?	string	<i>@ProductTypeDetails</i> specifies additional details of the product or product part that MAY be site specific and MAY be human readable. <i>@ProductType</i> SHOULD be present if <i>@ProductTypeDetails</i> is specified.
<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 intents.

4.1.2 Product Amount

Product/*@Amount* SHALL be applied within the context of one parent product. If **Product**/*@IsRoot="true"* then **Product**/*@Amount* SHALL specify the total number of products. If **Product**/*@IsRoot="false"* then **Product**/*@Amount* SHALL specify the total number of the respective child products required to create one parent product.

The following example shows the simplified description of 10 notebooks with a front and back cover and a 50 page book block.

Example 4.1: Amounts in a Notebook

```
<ProductList>
  <Product Amount="10" IsRoot="true" ProductType="Notebook">
    <Intent Name="BindingIntent">
      <BindingIntent BindingSide="Top" BindingType="EdgeGluing" ChildRefs="IBack IBody ICover"/>
    </Intent>
  </Product>
  <Product Amount="1" ID="ICover" IsRoot="false" ProductType="FrontCover"/>
  <Product Amount="50" ID="IBody" IsRoot="false" ProductType="BookBlock"/>
  <Product Amount="1" ID="IBack" IsRoot="false" ProductType="BackCover"/>
</ProductList>
```

4.1.2.1 Product Amount for Variable Data

If a **Product** contains a **VariableIntent**, then **Product/@Amount** SHALL refer to the number of instance documents, also referred to as recipients or records.

The following example describes a variable job with two finishing options. The entire job has 10000 records of which 9000 are brochures and 1000 are hardcover books. Each brochure and each book has one cover and one body.

Example 4.2: Amounts in Variable Data

```
<ProductList>
  <Product Amount="10000" IsRoot="true">
    <Intent Name="VariableIntent">
      <VariableIntent ChildRefs="IDBrochure IDBook" VariableType="Area"/>
    </Intent>
  </Product>
  <Product Amount="1000" ID="IDBook" IsRoot="false" ProductType="Book">
    <Intent Name="BindingIntent">
      <BindingIntent BindingType="HardCover" ChildRefs="IDBookCover IDBody"/>
    </Intent>
  </Product>
  <Product Amount="1" ID="IDBookCover" IsRoot="false" ProductType="Cover"/>
  <Product Amount="1" ID="IDBody" IsRoot="false"/>
  <Product Amount="9000" ID="IDBrochure" IsRoot="false">
    <Intent Name="BindingIntent">
      <BindingIntent BindingType="SaddleStitch" ChildRefs="IDBrochureCover IDBody"/>
    </Intent>
  </Product>
  <Product Amount="1" ID="IDBrochureCover" IsRoot="false" ProductType="Cover"/>
</ProductList>
```

4.1.3 Intent

Intent elements are a container for specific **Product Intent** elements. Product elements SHALL contain at most one intent element with the same **Intent/@Name**. If multiple product parts with different intent descriptions are needed, each product part SHALL be defined as a separate **Product**.

Table 4.2: Intent Element

<i>DescriptiveName</i> ?	string	Human readable descriptive name of the Intent .
<i>ExternalID</i> ?	NMTOKEN	Identifier of the Intent in an external system such as an MIS.
<i>Name</i>	NMTOKEN	Type of the product intent. A list of predefined intent types is specified in ▶ Table 4.3 Product Intent Elements. Extension intent types MAY be defined. See ▶ Section 3.5.4 Creating Extension Intent Elements for details.
Product Intent ?	element	Details of the Intent . The XML element name SHALL be the value of @Name .
<foreign namespace elements> *	element	Any elements in a foreign namespace. Foreign namespace extensions SHOULD NOT duplicate functionality of XJDF .

4.1.4 Product Intent

A **Product Intent** is any specific intent defined in this chapter (e.g. **BindingIntent**). A **Product Intent** is a child of a **Intent** element.

The following table defines the list of product intents.

Table 4.3: Product Intent Elements

NAME	DESCRIPTION
AssemblingIntent ?	This intent specifies the placing or inserting of one component within another, using information that identifies page location, position and attachment method.
BindingIntent ?	This intent specifies the binding intent for a Product .
ColorIntent ?	This intent specifies the type of ink to be used for a Product .
ContentCheckIntent ?	This intent specifies the prepress proofing intent for a Product , using information that identifies the type, quality, brand name and overlay of the proof.
EmbossingIntent ?	This intent specifies the embossing and/or foil stamping intent for a Product .
FoldingIntent ?	This intent specifies the fold intent for a Product using information that identifies the number of folds, the height and width of the folds, and the folding catalog number.
HoleMakingIntent ?	This intent specifies the hole making intent for a Product .
LaminatingIntent ?	This intent specifies the laminating intent for a Product using information that identifies whether or not the product is laminated.
LayoutIntent ?	This intent records the size of the finished pages for the product component.
MediaIntent ?	This intent describes the media to be used for the product component.
ProductionIntent ?	This intent specifies the manufacturing intent and considerations for a Product using information that identifies the desired result or specified manufacturing path.
ShapeCuttingIntent ?	This intent specifies form and line cutting for a Product .
VariableIntent ?	This intent specifies the variations for printed data with variable content.

4.1.5 Representation of Product Binding

[BindingIntent](#) and [AssemblingIntent](#) SHALL specify how multiple product parts are combined.

4.2 AssemblingIntent

This [Product Intent](#) element specifies the creation of a composite component by providing page location, position and attachment method of the respective child products that shall be assembled with the parent product. The containing [Product](#) SHALL be referenced in [AssemblingIntent](#)/[@Container](#).

Note: The containing [Product](#) is not identical to this parent [Product](#). For instance an empty envelope (the product that is referenced by [@Container](#)) is not the same thing as a filled envelope (the parent [Product](#)). Whereas products that are bound together with [BindingIntent](#) SHALL be counted when calculating the page numbers of final bound products, [AssemblyItems](#) SHALL be ignored when calculating page numbers.

Intent Properties

Process Resource Pairing: [InsertingParams](#)

Table 4.4: AssemblingIntent Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
Container	IDREF	@Container SHALL reference the main Product that the additional products that are referenced from AssemblyItem , BindIn , BlowIn or StickOn are assembled with. @Container SHALL NOT reference the parent Product of this AssemblingIntent .
AssemblyItem *	element	Each AssemblyItem element describes an individual item that is assembled with the main Product that is referenced by @Container .
BindIn *	element	Each BindIn element describes an individual insert that is glued into the main Product that is referenced by @Container .
BlowIn *	element	Each BlowIn element describes an individual insert that is loosely inserted into the main Product that is referenced by @Container .

Table 4.4: *AssemblingIntent Element (Sheet 2 of 2)*

NAME	DATA TYPE	DESCRIPTION
<i>StickOn</i> *	element	Each <i>StickOn</i> element describes an individual child <i>Product</i> that is glued onto the main <i>Product</i> that is referenced by <i>@Container</i> . <i>StickOn</i> is typically used for labels.

4.2.1 AssemblyItem

An *AssemblyItem* element describes any individual item that is assembled with the main *Product*. Examples of assembly items include stands for roll-up displays or frames.

Figure 4-1: Roll-up display

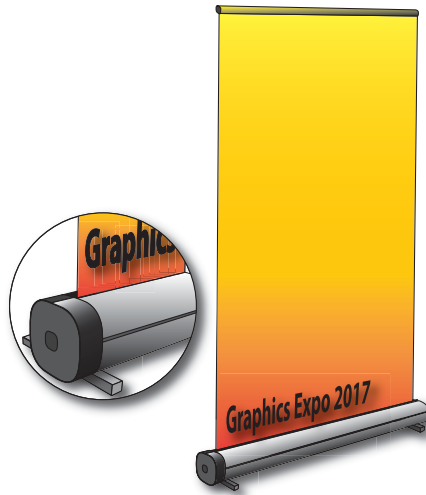


Table 4.5: *AssemblyItem Element*

NAME	DATA TYPE	DESCRIPTION
<i>ChildRef</i>	IDREF	<i>Product</i> / <i>@ID</i> of the product that describes the <i>AssemblyItem</i> .

4.2.2 BindIn

BindIn elements describe inserts that are glued into the main product.

Table 4.6: *BindIn Element*

NAME	DATA TYPE	DESCRIPTION
<i>ChildRef</i>	IDREF	<i>Product</i> / <i>@ID</i> of the product that describes the insert.
<i>Folio</i> ?	integer	Index of the parent surface where the insert SHALL be bound in the context of the main product that is referenced by <i>AssemblingIntent</i> / <i>@Container</i> .
<i>Orientation</i> ?	enumeration	Orientation of the insert in the coordinate system of the surface specified by <i>@Folio</i> . Allowed value is from: ▶ Orientation.
<i>Position</i> ?	XYPair	Position of the bottom left corner of the insert in the coordinate system of the surface specified by <i>@Folio</i> after applying all rotations.
<i>Glue</i> ?	element	Details of the glue used to fasten the insert.

4.2.3 BlowIn

BlowIn elements describe inserts that are loosely inserted into the main product. This includes filling items into an envelope.

Table 4.7: BlowIn Element

NAME	DATA TYPE	DESCRIPTION
<i>ChildRef</i>	IDREF	Product / <i>@ID</i> of the product that describes the insert.
<i>FolioFrom</i> ?	integer	Index of the first valid parent surface where the insert SHALL be placed in the context of the main product that is referenced by AssemblingIntent / <i>@Container</i> .
<i>FolioTo</i> ?	integer	Index of the last valid parent surface where this Product SHALL be placed in the context of the main product that is referenced by AssemblingIntent / <i>@Container</i> .
<i>Orientation</i> ?	enumeration	Orientation of the referenced insert in the coordinate system of the surface specified by <i>@Folio</i> . Allowed value is from: ▶ Orientation.

Example 4.3: Inserting a Letter into an Envelope

This example illustrates using **BlowIn** to describe a single letter in an envelope.

```
<ProductList>
  <Product Amount="10" IsRoot="true" ProductType="FilledEnvelope">
    <Intent Name="AssemblingIntent">
      <AssemblingIntent Container="ID_Envelope">
        <BlowIn ChildRef="ID_Letter"/>
      </AssemblingIntent>
    </Intent>
  </Product>
  <Product Amount="1" ExternalID="MISID_Envelope" ID="ID_Envelope"
    IsRoot="false" ProductType="Envelope"/>
  <Product Amount="1" ID="ID_Letter" IsRoot="false" ProductType="Letter"/>
</ProductList>
```

4.2.4 StickOn

StickOn elements describe labels that are applied to the main product.

Table 4.8: StickOn Element

NAME	DATA TYPE	DESCRIPTION
<i>ChildRef</i>	IDREF	Product / <i>@ID</i> of the product that describes the stick-on.
<i>Face</i> ?	enumeration	Location of the stick-on on the Product . <i>@Face</i> SHALL NOT be specified if <i>@Folio</i> is specified. Values are from: ▶ Face.
<i>Folio</i> ?	integer	Index of the parent surface where the stick-on SHALL be placed in the context of the main product that is referenced by AssemblingIntent / <i>@Container</i> . <i>@Folio</i> SHALL NOT be specified if <i>@Face</i> is specified.
<i>Orientation</i> ?	enumeration	Orientation of the stick-on in the coordinate system of the surface specified by <i>@Folio</i> or <i>@Face</i> . Allowed value is from: ▶ Orientation.
<i>Position</i> ?	XYPair	Position of the bottom left corner of the stick-on in the coordinate system of the surface specified by <i>@Folio</i> or <i>@Face</i> after applying all rotations.
<i>Glue</i> ?	element	Details of the glue used to fasten the stick-on.

4.3 BindingIntent

This **Product Intent** specifies the binding intent for a **Product** using information that identifies the desired type of binding and which sides SHALL be bound. All other **Products** SHALL be bound in the order of their appearance in **BindingIntent**/

PRODUCT INTENT

@ChildRefs. When stack binding (see **Gathering**) the first product in the **BindingIntent**/**@ChildRefs** list SHALL represent the bottom or back of the bound items (and therefore the last product SHALL represent the top or front). When wrap around binding (see **Collecting**) the first product in the **BindingIntent**/**@ChildRefs** list SHALL represent the outermost item of the bound items (and therefore the last product SHALL represent the innermost item).

Intent Properties

Process Resource Pairing: **BlockPreparationParams**, **CaseMakingParams**, **CasingInParams**, **CoverApplicationParams**, **EndSheetGluingParams**, **GluingParams**, **InsertingParams**, **JacketingParams**, **LooseBindingParams**, **SpinePreparationParams**, **SpineTapingParams**, **StitchingParams**, **ThreadSealingParams**, **ThreadSewingParams**

Table 4.9: BindingIntent Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>BackCoverColor</i> ?	enumeration	Defines the color of the back cover material of the binding. Allowed value is from: ▶ NamedColor.
<i>BackCoverColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If @BackCoverColorDetails is supplied, @BackCoverColor SHOULD also be supplied.
<i>BindingColor</i> ?	enumeration	Defines the color of the spine material of the binding. Allowed value is from: ▶ NamedColor.
<i>BindingColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If @BindingColorDetails is supplied, @BindingColor SHOULD also be supplied.
<i>BindingOrder</i> ?	enumeration	Specifies whether the child Component resources are to be collected or gathered if multiple child Component resources are combined. Allowed values are: None – The products referenced by @ChildRefs are NOT bound together. Typically used for flatwork jobs. Collecting – The products referenced by @ChildRefs are collected on a spine and placed within one another. The first Component is on the outside. Gathering – The child Component resources are gathered on a pile and placed on top of one another. The first child product specified by @ChildRefs is on the top.
<i>BindingSide</i> ?	enumeration	@BindingSide indicates which side of the product SHALL be bound. Each of these values SHALL identify the binding edge. @BindingSide is defined in the coordinate system of the product. @BindingSide SHALL NOT be provided if @BindingOrder="None" . Allowed value is from: ▶ Edge.
<i>BindingType</i>	enumeration	Describes the desired binding for the job. Allowed value is from: ▶ BindingType.
<i>ChildRefs</i> ?	IDREFS	Reference to one or more child products each identified by Product / @ID (e.g., cover and body of a book) that SHALL be bound together.
<i>CoverColor</i> ?	enumeration	Defines the color of the cover material of the binding. Allowed value is from: ▶ NamedColor.
<i>CoverColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If @CoverColorDetails is supplied, @CoverColor SHOULD also be supplied.
<i>AdhesiveNote</i> ?	element	Details of AdhesiveNote binding. AdhesiveNote SHALL NOT be provided unless BindingIntent / @BindingType="AdhesiveNote" .
<i>EdgeGluing</i> ?	element	Details of EdgeGluing . EdgeGluing SHALL NOT be provided unless BindingIntent / @BindingType="EdgeGluing" .
<i>HardCoverBinding</i> ?	element	Details of HardCoverBinding . HardCoverBinding SHALL NOT be provided unless BindingIntent / @BindingType="HardCover" .
<i>LooseBinding</i> ?	element	Details of LooseBinding . LooseBinding SHALL NOT be provided unless BindingIntent / @BindingType is one of "ChannelBinding" , "CoilBinding" , "CombBinding" , "RingBinding" or "StripBinding" .

Table 4.9: BindingIntent Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
SaddleStitching ?	element	Details of SaddleStitching . SaddleStitching SHALL NOT be provided unless BindingIntent/@BindingType="SaddleStitch" .
SideStitching ?	element	Details of SideStitching . SideStitching SHALL NOT be provided unless BindingIntent/@BindingType="SideStitch" .
SoftCoverBinding ?	element	Details of SoftCoverBinding . SoftCoverBinding SHALL NOT be provided unless BindingIntent/@BindingType="SoftCover" .
Tabs ?	element	Details of Tabs .

4.3.1 AdhesiveNote

Details of adhesive note binding.

Table 4.10: AdhesiveNote Element

NAME	DATA TYPE	DESCRIPTION
Glue ?	element	Glue provides details of the shape of the glue application and type of glue used.

4.3.2 EdgeGluing

Table 4.11: EdgeGluing Element

NAME	DATA TYPE	DESCRIPTION
EdgeGlue ?	enumeration	Glue type used to glue the edge of the gathered sheets. Allowed value is from: ▶ Glue .

4.3.3 HardcoverBinding

Table 4.12: HardcoverBinding Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
BlockThreadSewing ?	boolean	Specified if the block is thread sewn.
CoverStyle ?	NMTOKEN	Defines the style of the cover board. Values include: Simple – Single layer cover board, see ▶ Figure 4-2: Structure of a normal hardcover book. Padded – Padded cover board, see ▶ Figure 4-3: Structure of a padded hardcover book.
EndSheets ?	boolean	@EndSheets SHALL be specified if end sheets SHALL be applied. Additional details of the end sheets MAY be specified by supplying a Product that is referenced by the parent BindingIntent/@ChildRefs with @ProductType="EndSheet" .
HeadBands ?	boolean	The following case binding choice specifies the use of head bands on a case bound book. If "true", head bands are inserted both top and bottom.
HeadBandColor ?	enumeration	Defines the color of the head band. Allowed value is from: ▶ NamedColor .
HeadBandColorDetails ?	string	A more specific, specialized or site-defined name for the color. If @HeadBandColorDetails is supplied, @HeadBandColor SHOULD also be supplied.

Table 4.12: HardCoverBinding Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Jacket</i> ?	enumeration	Specifies whether a hardcover jacket is needed and how it is attached. Details of the jacket MAY be described in the Product that is referenced by the parent BindingIntent / <i>@ChildRefs</i> whose <i>@ProductType</i> ="Jacket". Allowed values are: None – No jacket is needed. Loose – The jacket is loosely wrapped. Glue – The jacket is glued to the spine.
<i>JacketFoldingWidth</i> ?	float	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>SpineGlue</i> ?	enumeration	Glue type used to glue the book block to the cover. Allowed value is from: ▶ Glue.
<i>SpineOperations</i> ?	NMTOKENS	<i>@SpineOperations</i> lists the operations that SHOULD be performed when preparing the spine. Allowed values are from: ▶ Spine Operations.
<i>Thickness</i> ?	float	Specifies the thickness of the board that is wrapped as front and back covers of a case bound book, in points.
<i>TightBacking</i> ?	enumeration	Definition of the geometry of the back of the book block. Allowed value is from: ▶ TightBacking.
RegisterRibbon *	element	Number, materials, colors and details of register ribbons.

Figure 4-2: Structure of a normal hardcover book

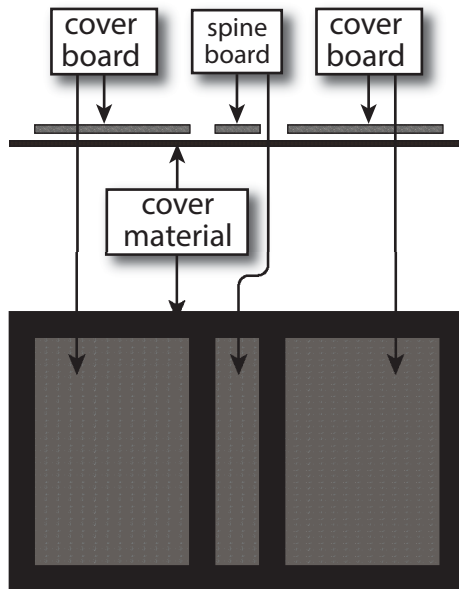
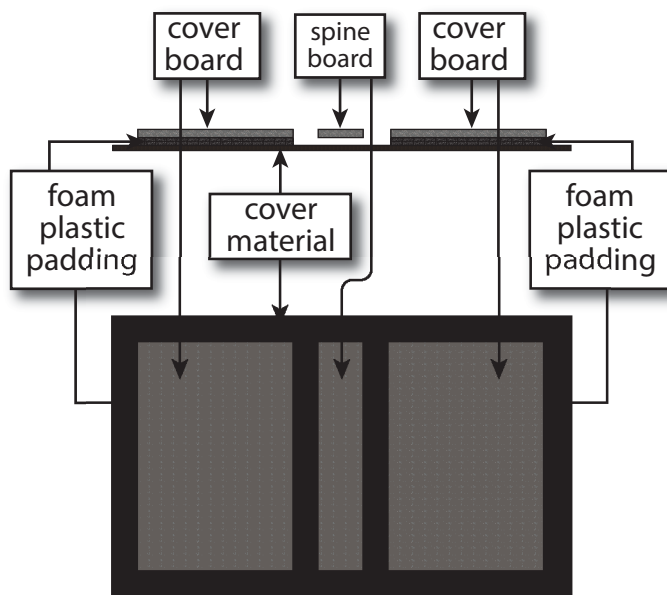


Figure 4-3: Structure of a padded hardcover book



4.3.4 LooseBinding

Table 4.13: LooseBinding Element

NAME	DATA TYPE	DESCRIPTION
<i>Brand</i> ?	string	@ <i>Brand</i> specifies the binder brand.
<i>Diameter</i> ?	float	@ <i>Diameter</i> specifies the diameter of coil, comb or rings, in points.
<i>HolePattern</i> ?	element	<i>HolePattern</i> 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.
<i>CoilBinding</i> ?	element	<i>CoilBinding</i> specifies additional details of coil binding. <i>CoilBinding</i> SHALL NOT be specified unless <i>BindingIntent</i> /@ <i>BindingType</i> ="CoilBinding".
<i>CombBinding</i> ?	element	<i>CombBinding</i> specifies additional details of either plastic comb binding or wire comb binding. <i>CombBinding</i> SHALL NOT be specified unless <i>BindingIntent</i> /@ <i>BindingType</i> ="CombBinding".
<i>RingBinding</i> ?	element	<i>RingBinding</i> specifies additional details of ring binding. <i>RingBinding</i> SHALL NOT be specified unless <i>BindingIntent</i> /@ <i>BindingType</i> ="RingBinding".

4.3.4.1 CoilBinding

Table 4.14: CoilBinding

NAME	DATA TYPE	DESCRIPTION
<i>CoilShape</i> ?	NMTOKEN	The shape of the wire coil used for the binding. Value includes those from: ▶ Comb and Coil Shapes.
<i>Material</i> ?	enumeration	The material available for forming the coil binding when <i>BindingIntent</i> /@ <i>BindingType</i> ="CoilBinding" or "WireComb". Allowed value is from: ▶ BinderMaterial.

4.3.4.2 CombBinding

Table 4.15: CombBinding

NAME	DATA TYPE	DESCRIPTION
<i>CombShape</i> ?	NMTOKEN	The shape of the plastic comb used for the binding. Value includes those from: ▶ Comb and Coil Shapes.
<i>Material</i> ?	enumeration	The material available for forming the comb binding when <i>BindingIntent</i> / <i>@BindingType</i> ="CombBinding". Allowed value is from: ▶ BinderMaterial.

4.3.4.3 RingBinding

Table 4.16: RingBinding

NAME	DATA TYPE	DESCRIPTION
<i>BinderMaterial</i> ?	NMTOKEN	<i>@BinderMaterial</i> describes the required material to be used for the binder cover. Values include: <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.
<i>RingShape</i> ?	NMTOKEN	<i>@RingShape</i> specifies the shape of the ring binder rings. Values include: <i>Round</i> <i>Oval</i> <i>D-shape</i> <i>SlantD</i>
<i>RivetsExposed</i> ?	boolean	<i>@RivetsExposed</i> describes the ring mechanism mounting in a 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.
<i>ViewBinder</i> ?	NMTOKEN	<i>@ViewBinder</i> specifies the details of clear vinyl outer-wrap types on top of a colored base wrap: Values include: <i>Embedded</i> – Printed material is embedded by sealing between the colored and clear vinyl layers during the binder manufacturing. <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.

4.3.5 SaddleStitching

Table 4.17: SaddleStitching Element

NAME	DATA TYPE	DESCRIPTION
<i>StapleShape</i> ?	enumeration	Specifies the shape of the staples to be used. Allowed value is from: ▶ StapleShape.
<i>StitchNumber</i> ?	integer	Number of stitches used for saddle stitching.

4.3.6 SideStitching

Table 4.18: SideStitching Element

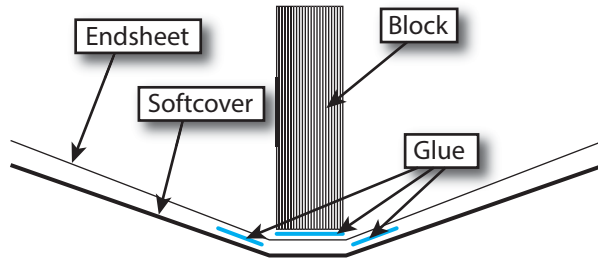
NAME	DATA TYPE	DESCRIPTION
<i>StapleShape</i> ?	enumeration	Specifies the shape of the staples to be used. Allowed value is from: ▶ StapleShape.
<i>StitchNumber</i> ?	integer	Number of stitches used for side stitching.

4.3.7 SoftCoverBinding

Table 4.19: SoftCoverBinding Element

NAME	DATA TYPE	DESCRIPTION
<i>BlockThreadSewing</i> ?	boolean	Specifies whether the block is also thread sewn.
<i>EndSheets</i> ?	boolean	@EndSheets SHALL be specified if end sheets SHALL be applied. Additional details of the end sheets MAY be specified by supplying a <i>Product</i> that is referenced by the parent <i>BindingIntent</i> / <i>@ChildRefs</i> whose <i>@ProductType</i> ="EndSheet".
<i>FoldingWidth</i> ?	float	Definition of the dimension of the folding width of the front cover fold. See <i>JacketingParams</i> for details.
<i>FoldingWidthBack</i> ?	float	Definition of the dimension of the folding width of the back cover fold. If not specified, <i>@FoldingWidthBack</i> defaults to <i>@FoldingWidth</i> .
<i>GlueProcedure</i> ?	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 end sheets but not at the spine. "SideOnly" books are also referred to as "layflat" if <i>@EndSheets</i> is also specified. See ▶ Figure 4-4: Structure of a book with <i>GlueProcedure</i> = "SideOnly" (Layflat). <i>SingleSide</i> – Swiss brochure. <i>SideSpine</i> – Both side gluing and spine gluing.
<i>Scoring</i> ?	enumeration	Scoring option for <i>SoftCoverBinding</i> . Values are based on viewing the cover in its flat, pre-bound state. Allowed values are: <i>TwiceScored</i> <i>QuadScored</i> <i>None</i>
<i>SpineGlue</i> ?	enumeration	Glue type used to glue the book block to the cover. Allowed value is from: ▶ Glue.
<i>SpineOperations</i> ?	NMTOKENS	@ <i>SpineOperations</i> lists the operations that SHOULD be performed when preparing the spine. Allowed values are from: ▶ Spine Operations.

Figure 4-4: Structure of a book with GlueProcedure = "SideOnly" (Layflat)



4.3.8 Tabs

Specifies tabs in a bound document.

Table 4.20: Tabs Element

NAME	DATA TYPE	DESCRIPTION
<i>ReinforceTabs</i> ?	boolean	If "true", the tab extension will be reinforced, e.g. with polyester film.
<i>ReinforceBind</i> ?	boolean	If "true", the tab bind edge will be reinforced, e.g. with polyester film.
<i>ReinforceColor</i> ?	enumeration	Specifies the color of the tab extension reinforcement. <i>@ReinforceColor</i> SHALL NOT be specified unless <i>@ReinforceTabs</i> ="true". Allowed value is from: ▶ NamedColor.
<i>ReinforceColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If <i>@ReinforceColorDetails</i> is supplied, <i>@ReinforceColor</i> SHOULD also be supplied.
<i>TabBrand</i> ?	string	Strings providing available brand names for the <i>Tabs</i> .
<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.
<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> ?	float	Distance tab extends beyond the body of the book block, in points.
<i>TabBodyCopy</i> ?	boolean	If "true", color will be applied not only on tab extension, but also on tab body. Note: The lack of body copy allows all tabs within a bank to be printed on a single sheet.

4.4 ColorIntent

ColorIntent specifies the color and varnishing of the product. Each surface SHALL be specified individually in a **SurfaceColor** element. Single sided printing SHALL be specified by providing exactly one **SurfaceColor** element.

In addition to the printed images, **ColorIntent** also provides details of protective or gloss enhancing coatings. Customers may either specify the performance characteristic they desire in the coating or specify a coating type. Common examples are water-resistance, and rub-resistance. Both characteristics may be required at the same time. An example is in the wine industry, where the white wine label has to survive transport rubbing, followed by water and rubbing ice cubes in a bucket upon serving.

Intent Properties

Process Resource Pairing: **Color**, **ColorantControl**, **ColorCorrectionParams**, **ColorSpaceConversionParams**, **Ink**, **VarnishingParams**

Table 4.21: ColorIntent Element

NAME	DATA TYPE	DESCRIPTION
SurfaceColor (Back) ?	element	SurfaceColor [@Surface="Back"] describes the color intent of the back surfaces of the final product. If not specified the "Back" surfaces SHALL NOT be marked.
SurfaceColor (Front) ?	element	SurfaceColor [@Surface="Front"] describes the color intent of the front surfaces of the final product. If not specified the "Front" surfaces SHALL NOT be marked.

4.4.1 SurfaceColor

This element specifies the color configuration of the desired product's surface.

Table 4.22: SurfaceColor Element

NAME	DATA TYPE	DESCRIPTION
Coatings ?	NMTOKENS	Material usually applied to a full surface on a press as a protective or gloss-enhancing layer over ink. Values include those from: ▶ Ink and Varnish Coatings. Note: Multiple NMTOKENS MAY be selected to indicate multiple coatings. Note: @Surface specifies the surface to which this coating applies. Note: Spot coating is specified in @ColorsUsed.
ColorsUsed ?	NMTOKENS	Array of colorant separation identifiers that are requested for this SurfaceColor . If specified @ColorsUsed SHALL contain a list of all separation identifiers used by the product or a list of spot colors specified by "Spot" that specifies a generic spot color whose details are unknown. "Spot" MAY be specified multiple times in one @ColorsUsed value. If not specified, then this SurfaceColor explicitly requests no colors on the surface that is specified in @Surface. If additional information about the colors and colorants is needed, it MAY be specified in ResourceSet/Resource/Color elements that are partitioned by matching Part/@Separation . In addition, partial (spot) coating MAY be specified by adding NMTOKENS with any value from @Coatings.
Coverage ?	float	Cumulative colorant coverage percentage. For example, a full sheet of 100% deep black in CMYK has @Coverage="400". Typical coverages based on one color plane are: Light – 1-9% Medium – 10-35% Heavy – 36+% Note: @Surface specifies the surface to which this coverage applies.
PrintStandard ?	NMTOKEN	Specifies the reference name of a characterization data set. See ▶ Appendix A.3.16 PrintStandard Characterization Data Sets for details.
Surface	enumeration	Allowed value is from: ▶ Side.

4.5 ContentCheckIntent

This [Product Intent](#) element specifies the prepress proofing and preflighting intent for a [Product](#).

Intent Properties

Process Resource Pairing: [ApprovalParams](#), [ApprovalDetails](#), [PreflightParams](#), [PreflightReport](#)

Table 4.23: ContentCheckIntent Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
PreflightItem *	element	PreflightItem defines the preflight rules for the Product .

Table 4.23: ContentCheckIntent Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ProofItem</i> *	element	Specifies the details of the proofs that are needed. If no <i>ProofItem</i> exists in a <i>ContentCheckIntent</i> , no customer proofs SHALL be provided. Note: <i>ProofItem</i> describes proofs that will be provided to the customer and does not specify internal production proofs.

4.5.1 PreflightItem

PreflightItem defines the preflight rules for the pages in a *Product*.

Table 4.24: PreflightItem Element

NAME	DATA TYPE	DESCRIPTION
<i>PreflightLevel</i> ?	enumeration	Level of content data checking / preflighting. The details are implementation specific. Allowed values are: <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. <i>Premium</i> – Highest available check for additional errors. This level MAY include manual repairs by the printer.

4.5.2 ProofItem

Table 4.25: ProofItem Element

NAME	DATA TYPE	DESCRIPTION
<i>Amount</i> ?	integer	Specifies the total number of copies of this proof that are needed.
<i>ColorType</i> ?	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.
<i>Contract</i> ?	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).
<i>HalfTone</i> ?	boolean	If <i>@HalfTone</i> ="true", the proof SHALL emulate halftone screens.
<i>ID</i> ?	ID	Identifier of the <i>ProofItem</i> . This field SHALL be specified if delivery of a proof is specified in <i>DeliveryParams</i> .
<i>PageIndex</i> ?	IntegerRange	Index of pages that SHALL be proofed in reader order. If <i>@PageIndex</i> is not specified, then all pages SHALL be proofed.
<i>ProofTarget</i> ?	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 SHALL be sent to the URL specified in <i>@ProofTarget</i> .

4.6 EmbossingIntent

This *Product Intent* specifies the embossing and/or foil stamping intent for a *Product* 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 4.26: EmbossingIntent Element

NAME	DATA TYPE	DESCRIPTION
<i>EmbossingItem</i> +	element	Each embossed image is described by one <i>EmbossingItem</i> .

4.6.1 EmbossingItem

Table 4.27: EmbossingItem Element

NAME	DATA TYPE	DESCRIPTION
<i>Direction</i> ?	enumeration	The direction of the image. Allowed value is from: ▶ EmbossDirection.
<i>EmbossingType</i>	enumeration	The embossing type required. Allowed value is from: ▶ EmbossType.
<i>Face</i> ?	enumeration	Position of the embossing on the product. Allowed value is from: ▶ Face.
<i>FoilColor</i> ?	enumeration	Defines the color of the foil material that is used for embossing. Allowed value is from: ▶ NamedColor.
<i>FoilColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If <i>@FoilColorDetails</i> is supplied, <i>@FoilColor</i> SHOULD also be supplied. <i>@FoilColorDetails</i> SHOULD be used to specify specialized foil properties such as holographic or transparent foils. Example combinations of <i>@FoilColor</i> and <i>@FoilColorDetails</i> include: Holographic foils: <i>@FoilColor</i> ="Silver" and <i>@FoilColorDetails</i> ="Holographic". Matte transparent foil: <i>@FoilColor</i> ="White" and <i>@FoilColorDetails</i> ="TransparentMatte".
<i>Height</i> ?	float	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 <i>@Direction</i> .
<i>ImageSize</i> ?	XYPair	The size of the bounding box of one single image.
<i>Position</i> ?	XYPair	Position of the lower left corner of the bounding box of the embossed image in the coordinate system of the surface of the <i>Component</i> that is selected by <i>@Face</i> .
<i>Separation</i> ?	NMTOKEN	<i>@Separation</i> identifies the separation within the PDL whose color values SHALL be used as the embossing values. A value of 0.0 in the PDL SHALL specify no embossing, a value of 1.0 in the PDL SHALL specify embossing with full depth. If a <i>ResourceSet/Resource/Color</i> element is specified for this separation, the value of <i>Color/@ColorType</i> SHALL be "DieLine".
<i>ToolName</i> ?	NMTOKEN	Name of the embossing tool.

4.7 FoldingIntent

This *Product Intent* specifies the straight line folding, creasing and perforating of a product.

Intent Properties

Process Resource Pairing: *CreasingParams*, *CuttingParams*, *Fold*, *FoldingParams*, *PerforatingParams*

Table 4.28: FoldingIntent Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>FoldCatalog</i> ?	NMTOKEN	Describes the folding scheme. 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. Values include those from: ▶ Fold Catalog.

Table 4.28: FoldingIntent 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>Orientation</i> ?	enumeration	@ <i>Orientation</i> indicates the orientation of the unfolded product with respect to the lay of the fold. A value of "Rotate0" SHALL be mapped to the lay of the fold on the lower left of the product prior to folding and the front side of the product oriented in the direction of an upward fold. Allowed value is from: ▶ Orientation.
<i>Crease</i> *	element	<i>Crease</i> elements describe 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>FoldCatalog</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>FoldCatalog</i> . <i>Fold</i> SHALL be specified if non-symmetrical folds are requested.
<i>Perforate</i> *	element	<i>Perforate</i> 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>FoldCatalog</i> is specified, the customer is requesting production perforation.

4.8 HoleMakingIntent

This *Product Intent* specifies the hole making intent for a *Product*. This *Product Intent* 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 4.29: HoleMakingIntent Element

NAME	DATA TYPE	DESCRIPTION
<i>HolePattern</i> +	element	Each <i>HolePattern</i> describes one hole or a specific set of holes that SHALL be provided. The coordinate system for applying the holes SHALL be the coordinate system of the <i>Product</i> .

4.9 LaminatingIntent

This *Product Intent* specifies the laminating intent for a *Product*.

Intent Properties

Process Resource Pairing: *LaminatingParams*

Table 4.30: LaminatingIntent Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Surface</i>	enumerations	The surface or surfaces to be laminated. Allowed values are from: ▶ Side.
<i>Temperature</i> ?	enumeration	Temperature used in the <i>Laminating</i> process. Allowed values are: Hot Cold
<i>Texture</i> ?	NMTOKEN	The intended texture of the laminate. Values include those from: ▶ Texture.

Table 4.30: LaminatingIntent Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Thickness</i> ?	float	Thickness of the laminating material. Measured in microns [µm].

4.10 LayoutIntent

This **Product Intent** 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 SHALL be imaged onto the finished media.

Intent Properties

Process Resource Pairing: **Assembly, BinderySignature, Layout**

Table 4.31: LayoutIntent Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Bleed</i> ?	float	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.
<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 D Media Size 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>Orientation</i> ?	enumeration	<i>@Orientation</i> SHALL specify the orientation of the artwork on the surface as defined by <i>@Sides</i> . <i>@Orientation</i> is used to define products such as back-lit displays, where the orientation of the image with respect to the final product is rotated or mirrored. Allowed value is from: ▶ Orientation.
<i>Pages</i> ?	integer	Specifies the number of finished pages (surfaces) of the product component, including blank pages. See <i>@SpreadType</i> for a discussion of the scope of <i>@Pages</i> . 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 was either saddle stitched or glued.
<i>Sides</i> ?	enumeration	<i>@Sides</i> specifies which side of the product SHALL be printed. Allowed value is from: ▶ Sides.

Table 4.31: LayoutIntent Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>SpreadType</i> ?	enumeration	<p>@<i>SpreadType</i> SHALL specify the treatment of individual PDF pages referenced by the DPart node for which @<i>SpreadType</i> is specified for imposition purposes.</p> <p>Allowed values are:</p> <p>SinglePage – The content of each page SHALL be imaged in a single cell in imposition. Each finished page SHALL be counted as an individual page. For instance, a booklet cover would have four pages.</p> <p>Spread – The content of each page SHALL be imaged as a single surface onto the final product. Examples include wraparound covers. Spread SHOULD NOT be provided for adjacent pages that are not imaged onto the same surface. Each surface of the spread SHALL be counted as an individual page. For instance, a wrap around cover would have two pages.</p> <p>Note: Products with finished pages of varying size such as wrap around covers with a spine or fold outs in a booklet will typically be defined as spreads.</p> <p>Note: Content will typically be provided as single pages.</p>

4.11 MediaIntent

This **Product Intent** describes the media to be used for the **Product**.

Intent Properties

Process Resource Pairing: **Media**

Table 4.32: MediaIntent Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>BackCoating</i> ?	enumeration	<p>@<i>BackCoating</i> SHALL specify the pre-process coating of the back surface of the media. If not specified the value of @<i>Coating</i> SHALL be applied.</p> <p>Allowed value is from: ▶ Coating.</p>
<i>BackISOPaperSubstrate</i> ?	enumeration	<p>@<i>ISOPaperSubstrate</i> SHALL specify the back surface of paper material defined in accordance with the print substrate set forth in ▶ [ISO12647-2:2013]. If not specified, the value of @<i>ISOPaperSubstrate</i> SHALL be applied.</p> <p>Allowed value is from: ▶ ISOPaperSubstrate.</p>
<i>Brand</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>BuyerSupplied</i> ?	boolean	Indicates whether the customer will supply the media.
<i>Coating</i> ?	enumeration	<p>@<i>Coating</i> SHALL specify the pre-process coating of the media.</p> <p>Allowed value is from: ▶ Coating.</p>
<i>Flute</i> ?	NMTOKEN	Single, capital letter that specifies the flute type of corrugated media. Values include those from: ▶ Flute Types.
<i>FluteDirection</i> ?	enumeration	Direction of the flute of corrugated media in the coordinate system of the product. Allowed value is from: ▶ MediaDirection.
<i>GrainDirection</i> ?	enumeration	Direction of the grain in the coordinate system of the Product . Allowed value is from: ▶ MediaDirection.
<i>ISOPaperSubstrate</i> ?	enumeration	<p>@<i>ISOPaperSubstrate</i> SHALL specify the type of paper material defined in accordance with the print substrate set forth in ▶ [ISO12647-2:2013].</p> <p>Allowed value is from: ▶ ISOPaperSubstrate.</p>
<i>LabColorValue</i> ?	LabColor	@ <i>LabColorValue</i> is the CIELAB color value of the media, computed as specified in ▶ [TAPPI T527].

Table 4.32: MediaIntent Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>MediaColor</i> ?	enumeration	Color of the media. If more-specific, specialized or site-specific media color names are needed, use @MediaColorDetails . Allowed value is from: ▶ NamedColor.
<i>MediaColorDetails</i> ?	string	A more specific, specialized or site-defined name for the media color. If @MediaColorDetails is supplied, @MediaColor SHOULD also be supplied. Note: There is a one-to-many relationship between entries in @MediaColor and @MediaColorDetails (e.g., @MediaColorDetails values of "Burgundy" and "Ruby" both correspond to a @MediaColor of "DarkRed").
<i>MediaQuality</i> ?	string	Named quality description of the media. Media with the same @MediaQuality 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.
<i>MediaType</i>	enumeration	Describes the medium being employed. Allowed value is from: ▶ MediaType.
<i>MediaTypeDetails</i> ?	NMTOKEN	Describes additional details of the medium described in @MediaType . Values include those from: ▶ MediaType Details. Note: Values from ▶ MediaType Details are RECOMMENDED. However, some process related values, such as "DryFilm", SHOULD NOT be used for this attribute.
<i>Opacity</i> ?	enumeration	The opacity of the media. Allowed value is from: ▶ Opacity.
<i>PrePrinted</i> ?	boolean	Indicates whether the media is preprinted.
<i>StockType</i> ?	NMTOKEN	@StockType defines the base size when calculating North American or Japanese paper weights. See ▶ Appendix C Media Weight for details including pre-defined values.
<i>Texture</i> ?	NMTOKEN	The intended texture of the media. Values include those from: ▶ Texture.
<i>Thickness</i> ?	float	The thickness of the chosen medium. Measured in microns [µm].
<i>Weight</i> ?	float	The intended weight of the media, measured in grammage (g/m ²) of the media. See ▶ Appendix C Media Weight for an explanation of how to calculate the US weight from the grammage for different stock types.
<i>Certification</i> *	element	Each Certification specifies a minimum requested paper certification level.

4.12 ProductionIntent

This [Product Intent](#) specifies the manufacturing intent and considerations for a [Product](#) using information that identifies the desired result or specified manufacturing path. If specific details of print quality, such as color quality, need to be specified, [@Types](#) SHOULD contain "QualityControl". A [QualityControlParams ResourceSet](#) that contains the requirements SHOULD also be provided.

Intent Properties

Table 4.33: ProductionIntent Element

NAME	DATA TYPE	DESCRIPTION
<i>PrintPreference</i> ?	enumeration	Intended result or goal. Allowed values are: Balanced – Request for a manufacturing process that balances the requirements for cost, speed and quality. CostEffective – Request for the most cost effective manufacturing process. Fastest – Request for the most time effective manufacturing process. Cost and quality can be sacrificed for a fast turnaround time. HighestQuality – Request for the manufacturing process that will result in the highest quality.
<i>PrintProcess</i> ?	NMTOKENS	Print process requested. If more than one value is specified, then <i>@PrintProcess</i> requests hybrid printing, e.g. inkjet imprint on a preprinted shell. Values include those from: ▶ Printing Technologies.

4.13 ShapeCuttingIntent

ShapeCuttingIntent describes finishing of products with irregular shapes, including die cutting and adding windows to envelopes.

Intent Properties

Process Resource Pairing: *CuttingParams, ShapeCuttingParams*

Table 4.34: ShapeCuttingIntent Element

NAME	DATA TYPE	DESCRIPTION
<i>ShapeCut</i> +	element	Array of all <i>ShapeCut</i> elements. Used when each shape is exactly specified.

4.13.1 ShapeCut

Table 4.35: ShapeCut Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>CutBox</i> ?	rectangle	Specification of a rectangular window. An orthogonal line MAY be defined by specifying a rectangle with identical dimensions.
<i>CutDepth</i> ?	enumeration	Allowed values are: Full – The form is completely cut out or perforated. Partial – The form is not completely cut out or perforated. The exact depth MAY be specified in <i>ShapeCuttingParams</i> .
<i>CutOut</i> ?	boolean	<i>@CutOut</i> specifies whether the inside or outside of the <i>ShapeCut</i> SHALL be removed. If <i>@CutOut="true"</i> , the inside of a specified shape SHALL be removed, otherwise the outside of a specified shape SHALL 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.
<i>CutType</i> ?	enumeration	Type of cut or perforation used. Allowed values are: Cut – Full cut. Perforate – Interrupted perforation that does not span the entire sheet.

Table 4.35: ShapeCut Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ShapeType</i>	enumeration	Describes any precision cutting other than hole making. Allowed values are: Line – The coordinates specified in <i>@CutBox</i> specify the end points of a straight line. Path – Any irregular shape. Additional details SHOULD be provided in <i>@CutPath</i> or <i>@ShapeTypeDetails</i> . Rectangular – The coordinates specified in <i>@CutBox</i> specify the lower left and upper right coordinates of a rectangle. Round – Circular or elliptical shape depending on the aspect ratio of <i>@CutBox</i> . RoundedRectangle – Rectangle with rounded corners. The coordinates specified in <i>@CutBox</i> specify the outer bounds of the rectangle.
<i>ShapeTypeDetails</i> ?	string	A more specific, specialized or site-defined name for the shape of the <i>ShapeCut</i> .

4.14 VariableIntent

VariableIntent specifies the variations of the content for printed data with variable content such as lottery tickets or direct mail.

Intent Properties

Process Resource Pairing: *DigitalPrintingParams*, *LayoutElementProductionParams*

Table 4.36: VariableIntent Element

NAME	DATA TYPE	DESCRIPTION
<i>Area</i> ?	float	Ratio of the document that can contain variable content. A value of 0 specifies a non variable document. A value of 1 specifies a full variable document.
<i>AveragePages</i> ?	integer	<i>AveragePages</i> SHALL specify the average number of printed pages in each record.
<i>ChildRefs</i> ?	IDREFS	<i>Product</i> / <i>@ID</i> of the product elements that describe individual finishing variants. <i>@ChildRefs</i> SHALL NOT be specified if <i>AssemblingIntent</i> or <i>BindingIntent</i> are specified for this product.
<i>ColorsUsedBack</i> ?	NMTOKENS	Array of colorant separation identifiers that are required to print the variable part of the documents. The values that are specified in <i>@ColorsUsedBack</i> SHALL also be specified in <i>ColorIntent/SurfaceColor</i> [<i>@Surface</i> ="Back"]/ <i>@ColorsUsed</i> . See <i>ColorIntent/SurfaceColor</i> / <i>@ColorsUsed</i> for additional details.
<i>ColorsUsedFront</i> ?	NMTOKENS	Array of colorant separation identifiers that are required to print the variable part of the documents. The values that are specified in <i>@ColorsUsedFront</i> SHALL also be specified in <i>ColorIntent/SurfaceColor</i> [<i>@Surface</i> ="Front"]/ <i>@ColorsUsed</i> . See <i>ColorIntent/SurfaceColor</i> / <i>@ColorsUsed</i> 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> .
<i>MinPages</i> ?	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> .
<i>NumberOfCopies</i> ?	integer	Average number of copies of each record. This value SHALL equal "1" for fully variable data.
<i>VariableType</i>	enumeration	Type of variable content. Allowed Values are (in order of rising complexity): OneLine - A single line of text data is variable. OneLine includes simple numbering applications. AddressField - Multiple lines of text data are variable. IdentificationField - The variable data includes a Barcode or QR-Code. Area - The area as defined in <i>@Area</i> is fully variable.

Table 4.36: VariableIntent Element

NAME	DATA TYPE	DESCRIPTION
<i>VariableQuality</i> ?	enumeration	<p>@<i>VariableQuality</i> specifies the desired quality of the variable data.</p> <p>Allowed Values are:</p> <p>Simple - The variable text MAY be recognized as printed by a different technology such as dot matrix or simple inkjet overprints.</p> <p>Imprint - The variable data SHOULD be similar to the non-variable part but MAY be imprinted.</p> <p>Full - All data SHOULD be printed with the same technology.</p>

5 Processes

The following chapter lists the individual processes that are defined in detail for **XJDF**.

5.1 Process Template

Processes are defined by their input and output resources (i.e. **ResourceSet**[@Usage="Input"] and **ResourceSet**[@Usage="Output"]). The requirements for the individual processes are provided in the tables below. ▶ Table 5.1 Generic Input ResourceSets provides a list of resources that are valid for any process. In addition to the resources listed in the tables for each process, extension **ResourceSets**, i.e. those in a foreign namespace, MAY be provided. Foreign namespace extensions SHOULD NOT duplicate any **XJDF** functionality. See ▶ Creating Extension ResourceSets for details.

Note: The cardinality requirements for **ResourceSet** and **Intent** elements, which are defined in this chapter and can be derived from the value of **XJDF**/@Types, are not validated by the XML schema provided by **CIP4**.

Note: In this chapter, each entry in the 'Name' column of a table provides the requirements in the format

- Name: The **ResourceSet**/@Name of the resource, e.g. **Media** or **RunList**.
- (ProcessUsageValue): If present, the **ResourceSet**/@ProcessUsage of the resource, e.g. **Document** or **Marks** in case of a **RunList**.
- Cardinality: The cardinality of the **ResourceSet**. See ▶ Table 1.2 Cardinality Symbols for details. The cardinality applies to the number of **ResourceSet** elements. Each **ResourceSet** may contain multiple **Resource** elements.

Table 5.1: Generic Input ResourceSets (Sheet 1 of 2)

NAME	DESCRIPTION
ApprovalDetails ?	ApprovalDetails MAY be appended to processes in order to model proofing and verification requirements. If multiple approvals are requested for an individual workstep, ApprovalDetails SHALL be partitioned by Part /@Option. For more information on the Approval process, see ▶ Section 5.3.1 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 ?	List of internal and external contacts that are associated with processing this XJDF . Resource/Part /@ContactType SHALL be provided for all contacts.
CustomerInfo ?	CustomerInfo (without the @ProcessUsage attribute) 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 XJDF .
MiscConsumable *	Generic consumables that are associated with processing this XJDF . If multiple MiscConsumable are specified, ResourceSet /@ProcessUsage SHALL be specified. The preferred value of ResourceSet /@ProcessUsage for a process specific MiscConsumable is provided in that process's table of input resources shown below. Additional MiscConsumable MAY be specified, in which case the value of ResourceSet /@ProcessUsage SHOULD be set to the value of MiscConsumable /@Type.
NodeInfo ?	NodeInfo (without the @ProcessUsage attribute) specifies scheduling information about the explicit process described by this XJDF .
NodeInfo (EndCustomer) ?	NodeInfo (EndCustomer) specifies scheduling information from the end customers in a subcontracting situation where the direct customer is not the end customer.
Preview ?	Any number of previews MAY be associated with a process and used for display purposes such as illustrating the orientation of a resource to an operator. For details of coordinate systems see ▶ Section 2.6 Coordinate Systems in XJDF. Part /@PreviewType SHOULD be "Thumbnail" or "Viewable".

Table 5.1: Generic Input ResourceSets (Sheet 2 of 2)

NAME	DESCRIPTION
Tool *	Miscellaneous reusable tools required for a process. If multiple tools of different types are specified, ResourceSet / @ProcessUsage SHALL be set to the value of Tool / @ToolType .
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. If multiple counters are provided, UsageCounter SHALL be partitioned by Option and Part / @Option SHOULD contain a value from UsageCounter / @CounterTypes .
<foreign namespace ResourceSet > *	Any ResourceSet in a foreign namespace. Foreign namespace extensions SHOULD NOT duplicate functionality of XJDF . See ▶ Creating Extension ResourceSets for details.

5.2 Combining Individual Process Steps

The processes described in this chapter define individual workflow steps that are assumed to be executed by a single-purpose device. Some controllers and devices are able to combine the functionality of multiple single-purpose devices and execute more than one process type. For example, a digital printer might be able to execute the **Interpreting**, **Rendering** and **DigitalPrinting** processes. Each **XJDF** SHALL contain a **@Types** attribute, which in turn contains an ordered list of values of each of processes that the **XJDF** 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.

Example 5.1: Combined Process Steps

Example of combining three processes in sequence: **Interpreting**, **Rendering** and **DigitalPrinting**.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="CombinedExample" Types="Interpreting Rendering DigitalPrinting">
</XJDF>
```

5.2.1 Exchange ResourceSets in combined processes

A **ResourceSet** that is produced by one process and immediately consumed by the following process NEED NOT be explicitly specified in the **XJDF**. Any such **ResourceSet** MAY be completely under the control of the receiving device. In the example above, the output **RunList** of the **Interpreting** process is the input **RunList** of **Rendering** process and NEED NOT be specified explicitly in the **XJDF**.

If an exchange **ResourceSet** is provided, **ResourceSet**/**@Usage** SHALL NOT be present and **ResourceSet**/**@CombinedProcessIndex** SHALL reference all processes that use the **ResourceSet** either as input or as output.

5.2.2 Usage of ResourceSets that are used as both input and output

Some processes, e.g. **Stripping** or **QualityControl**, modify a resource rather than consume or produce it.

Examples of modification include the expansion of a **Layout** by **Stripping** or the update of amounts in **Component** resources due to failing **QualityControl**. **ResourceSet**/**@Usage** of such a **ResourceSet** shall be set to the value where the **ResourceSet** is not used as an exchange resource as described in ▶ Section 5.2.1 Exchange ResourceSets in combined processes. If the **ResourceSet** is not used as an exchange **ResourceSet**, e.g. because **XJDF**/**@Types** is a single value, **ResourceSet**/**@Usage** shall be "Output".

Note: Printing and finishing processes do not modify **Component** resources; these processes have unique **Component** resources for their input and output because the resources are at different stages of production and thus need to be uniquely identifiable, e.g. for calculating the value of work in progress.

5.2.2.1 Example of input and combined ResourceSet

For an **XJDF** that combines stripping and imposition, i.e. where **XJDF**/**@Types**="Stripping Imposition", then **ResourceSet**[**@Name**="Layout"]/**@Usage**="Input" because **Layout** is an input to **Stripping** and an exchange **ResourceSet** between **Stripping** and **Imposition**.

5.2.2.2 Example of output and combined ResourceSet

For an **XJDF** that combines printing and quality control, i.e. where **XJDF**/**@Types**="ConventionalPrinting QualityControl", then **ResourceSet**[**@Name**="Component"]/**@Usage**="Output" because **Component** is an output of **QualityControl** and an exchange **ResourceSet** between **ConventionalPrinting** and **QualityControl**.

5.2.3 XJDF with Multiple Processes of the Same Type

XJDF/@Types MAY contain multiple instances of the same process type, e.g., "**Cutting Folding Cutting**". 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, ResourceSet/@CombinedProcessIndex** SHALL be present and refer to the index of the process type in the complete list of **XJDF/@Types**. In the example above, for instance **ResourceSet/@CombinedProcessIndex="0"** for the **CuttingParams** that apply to the first **Cutting** process and **ResourceSet/@CombinedProcessIndex="2"** for the **CuttingParams** that apply to the second **Cutting** process. **ResourceSet/@CombinedProcessIndex="1"** is not required for the **FoldingParams** since there is only one **Folding** process.

5.3 General Processes

General processes that can take place throughout the workflow.

5.3.1 Approval

The **Approval** process can take place at various steps in a workflow. For example, a **ResourceSet** (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. If **Approval** is combined with any other process type, the workstep that follows **Approval** in **XJDF/@Types** SHALL NOT commence until a successful **ApprovalDetails** is provided for a given partition.

Table 5.2: Approval – Input Resources

NAME	DESCRIPTION
ApprovalParams	Details of the approval process.
ResourceSet	The resources to be approved. When the input resource of an Approval process is a RunList that represents a ByteMap , it SHOULD be displayed on a viewing device.
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.3: Approval – Output Resources

NAME	DESCRIPTION
ApprovalDetails	Result of any Approval process given, for example, by a customer or foreman.
ResourceSet	This ResourceSet describes the resources after the Approval is complete.

5.3.2 Delivery

This process can be used to describe the delivery of an end product or a **ResourceSet** to or from a location that SHALL be specified in a **Contact** with **Part[@ContactType="Delivery"]**. Delivery of data over the network MAY also be specified in the **Delivery** process.

If the delivery only requires the address of a single end customer and no specific details of the delivery are known, then the delivery process need not be specifically parameterized.

Delivery to multiple destinations or in multiple steps SHOULD be specified in **DeliveryParams** that are partitioned by **@DropID**. If multiple **DeliveryParams** contain the same **@DropID**, they SHOULD be delivered in one delivery, regardless of whether the **DeliveryParams** belong to the same **XJDF** or not. Common delivery of multiple products to the same address SHALL be specified by providing multiple **DeliveryParams/DropItem** elements.

Table 5.4: Delivery – Input Resources

NAME	DESCRIPTION
Bundle ?	Bundle describes the structure of the packaging of the products that SHALL be delivered. If DeliveryParams is partitioned by Part/@DropID , Bundle SHOULD also be partitioned by Part/@DropID .
DeliveryParams	Details of the individual deliveries SHALL be provided in DeliveryParams .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.5: Delivery – Output Resources

NAME	DESCRIPTION
ResourceSet *	Any ResourceSet that is delivered to the process SHALL be specified as an output of the Delivery process.

Example 5.2: Split Delivery

The following example illustrates a split delivery of thirty books, ten of which go to the contact defined by "Drop1" and twenty of which go to the contact defined by "Drop2".

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="splitDelivery" Types="Product">
  <ProductList>
    <Product Amount="30" ID="IDBook" IsRoot="true" ProductType="Book"/>
  </ProductList>
  <ResourceSet Name="Contact" Usage="Input">
    <Resource>
      <Part ContactType="Delivery" DropID="Drop1"/>
      <Contact>
        <Address City="city1"/>
        <Person FirstName="Name1"/>
      </Contact>
    </Resource>
    <Resource>
      <Part ContactType="Delivery" DropID="Drop2"/>
      <Contact>
        <Address City="city2"/>
        <Person FirstName="Name2"/>
      </Contact>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="DeliveryParams" Usage="Input">
    <Resource>
      <Part DropID="Drop1"/>
      <DeliveryParams>
        <DropItem Amount="10" ItemRef="IDBook"/>
      </DeliveryParams>
    </Resource>
    <Resource>
      <Part DropID="Drop2"/>
      <DeliveryParams>
        <DropItem Amount="20" ItemRef="IDBook"/>
      </DeliveryParams>
    </Resource>
  </ResourceSet>
</XJDF>
```

5.3.3 ManualLabor

This process can be used to describe any process where resources are handled manually. The **ManualLabor** process is designed to monitor any type of non-automated labor from an MIS system.

Table 5.6: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.7: ManualLabor – Output Resources

NAME	DESCRIPTION
ResourceSet *	The resources that were created by manual work. In general these will be Component resources, but other resources MAY also be processed manually. If no output resources are specified, the ManualLabor process describes incidental work.

5.3.4 QualityControl

This process defines the setup and frequency of quality controls for a process. **QualityControl** is generally performed on **Component** resources.

Table 5.8: QualityControl – Input Resources

NAME	DESCRIPTION
QualityControlParams	Detailed definition of the QualityControl process.
ResourceSet	The resource to be quality controlled. In general this will be a Component .
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.9: QualityControl – Output Resources

NAME	DESCRIPTION
QualityControlResult	Results of the process, e.g. measurement statistics.
ResourceSet	This ResourceSet describes the resources after QualityControl has been applied.

5.3.5 Verification

The **Verification** process is used to confirm that a process has been completely executed.

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 5.10: Verification – Input Resources

NAME	DESCRIPTION
ResourceSet	The resources to be verified. The input will most often be a Component .
VerificationParams	Controls the verification requirements.
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.11: Verification – Output Resources

NAME	DESCRIPTION
ResourceSet	The resource after verification. Most often the ResourceSet will not be modified by Verification .
VerificationResult ?	Results of the process, e.g. measurement statistics.

5.4 Prepress Processes

This section lists all processes that are performed prior to printing. This includes processes that are performed to make digital assets press ready and the creation of physical assets such as plates or cut dies that are required for printing or converting.

5.4.1 Bending

The **Bending** device consumes a printing plate and bends and/or punches it. An in-line plate puncher SHOULD be modeled as a combined process consisting of **ImageSetting** and **Bending** processes.

Table 5.12: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.13: Bending – Output Resources

NAME	DESCRIPTION
ExposedMedia	The bent/punched ExposedMedia resource.

5.4.2 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** provides simple controls for adjusting colors. See **ColorSpaceConversion** for color manipulations based on ICC profiles.

Individual output color separations MAY be directly modified by providing an input [TransferCurve/@Curve](#) to the **ColorCorrection** process. If present [@Curve](#) shall be applied after any modifications specified in [ColorCorrectionParams](#).

Table 5.14: 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 SHALL be operated on.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.15: ColorCorrection – Output Resources

NAME	DESCRIPTION
RunList	List of color-corrected content elements.

5.4.3 ColorSpaceConversion

ColorSpaceConversion is the process of converting colors that are provided in a PDL to another 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 PDL.

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.

The target profile for color space conversion selection should be based on [ColorSpaceConversionParams/@ICCProfileUsage](#) in the following order of precedence.

- **UsePDL** – If present, the embedded target profile SHALL be used.
- **UseSupplied** – The embedded target profile SHALL NOT be used.

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 provide a DeviceLink¹ transform in [ColorSpaceConversionParams/ColorSpaceConversionOp/FileSpec](#)[@ResourceUsage="DeviceLinkProfile"]. The transform SHALL be applied when converting from a specific source color space to the final target device color space specified for the [ColorSpaceConversion](#) operation being applied. In these instances, the final target profile SHALL NOT be specified in [ColorSpaceConversionParams/FileSpec](#).

Table 5.16: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.17: ColorSpaceConversion – Output Resources

NAME	DESCRIPTION
RunList	List of pages, sheets or byte maps on which the selected operation has been performed.

5.4.4 DieDesign

This process describes the design of a die tool set with one or more stations starting from a [DieLayout](#) that describes the layout of the one-up designs on a die. The output of this process is a [DieLayout](#) resource, describing a tool set for the die cutter machine that can be used in a subsequent [DieMaking](#) process. [DieDesign](#) typically follows [DieLayoutProduction](#).

Table 5.18: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.19: DieDesign – Output Resources

NAME	DESCRIPTION
DieLayout +	A set of resources describing the die cutter tool set.

5.4.5 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 the positioning of the individual one-ups on the die. 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 [DieLayoutProduction](#) 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.

1. A DeviceLink transform is a transform that is defined in an ICC profile file (see ▶ [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 5.20: DieLayoutProduction – Input Resources

NAME	DESCRIPTION
DieLayoutProductionParams	The parameters for DieLayoutProduction .
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.21: DieLayoutProduction – Output Resources

NAME	DESCRIPTION
DieLayout	DieLayout describes a die cutter tool set. If DieLayoutProductionParams/@Estimate="True" , multiple alternative DieLayout Resource elements that SHALL be partitioned by Part/@Option SHALL be returned, otherwise a single DieLayout SHALL be generated.

Example 5.3: DieLayoutProduction: Single Shape and Two Sheet Sizes

Example of [DieLayoutProduction](#) of a single shape on 2 stock sheet sizes.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0"
  DescriptiveName="Single shape versus a set of sheet sizes"
  JobID="n_001003" JobPartID="ID234" Types="DieLayoutProduction">
  <ResourceSet ID="ShapeUp" Name="ShapeDef">
    <Resource>
      <ShapeDef>
        <FileSpec URL="file://myserver/myshare/olive.dd3"/>
      </ShapeDef>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="DieLayoutProductionParams" Usage="Input">
    <Resource>
      <DieLayoutProductionParams>
        <ConvertingConfig SheetHeightMax="2267.72"
          SheetHeightMin="2267.72" SheetWidthMax="2834.64" SheetWidthMin="2834.64"/>
        <ConvertingConfig SheetHeightMax="2834.64"
          SheetHeightMin="2834.64" SheetWidthMax="3401.57" SheetWidthMin="3401.57"/>
        <RepeatDesc AllowedRotate="None" GutterY="0" GutterY2="14.17"
          LayoutStyle="Reverse2ndRow" ShapeDefRef="ShapeUp"/>
      </DieLayoutProductionParams>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="DieLayout" Usage="Output"/>
</XJDF>
```

5.4.6 ImageEnhancement

The [ImageEnhancement](#) process describes generic image data processing.

Note: The source MAY be any image, but also text or vector graphics.

Table 5.22: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.23: ImageEnhancement – Output Resources

NAME	DESCRIPTION
RunList	List of page contents with images that have been manipulated as indicated by the ImageEnhancementParams resource.

5.4.7 ImageSetting

The **ImageSetting** process is executed by an imagesetter or platesetter that images a bitmap onto the film or plate media.

Table 5.24: ImageSetting – Input Resources

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.
ImageSetterParams ?	Controls the device specific features of the imagesetter.
Media	The film or plate prior to imaging.
RunList ?	Identifies the set of bitmaps to image. MAY contain bytemaps or images.
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.25: 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.

5.4.8 Imposition

The **Imposition** process is responsible for combining pages of input graphical content onto surfaces 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: The **Imposition** process specifies the task of combining pages and marks on sheets. The task of setting up the parameters needed for **Imposition** is defined by **Stripping**.

Table 5.26: 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 that are 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.27: Imposition – Output Resources

NAME	DESCRIPTION
RunList	The RunList represents a structured list of imposed surfaces.. Conceptually the output RunList will be partitioned by at least Part / @SheetName and Part / @Side to represent the individual printed surfaces. If the Imposition process is executed before any raster image processing, this will generally be consumed by an Interpreting process. In the case of where Imposition is executed after any raster image processing, 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**. 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)**) to specify the page content to be imposed.

In **XJDF**, there is a single **Layout** definition. When described fully (**@Automated = "false"**), the **Layout** resource partition structure explicitly defines the imposition to take place.

Note: 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 SHALL create the output **RunList**.

5.4.8.1 Execution Model for Automated Imposition

The **Imposition** process transforms the sequences of pages contained within the input **RunList** to a sequence of imposed sheet surfaces. The input **RunList (Document)** and the order of the documents defined by the **Layout** resource explicitly define the 'page to sheet' surface mapping transformation that SHALL be 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 set in the order specified in the input RunList (Document)
  For each Document in the order specified in the input RunList (Document)
    For each Layout partition that matches the Document
      For each Page in the Document
        Process the Page through the Layout partition

```

Thus, each **Resource** in the **ResourceSet[@Name="Layout"]** SHALL be processed in the XML order specified. Every document belonging to the current set is then evaluated against the partition keys specified for that **Layout** to determine if it SHALL be processed by that **Layout**.

The **RunList** output from the **Imposition** process represents a sequence of imposed sheet surfaces. The structure of the **Layout** 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 PDL metadata was in scope at the time the sheets were generated.

5.4.8.2 Cut and Stack Imposition

Pages are normally distributed onto an entire imposed sheet prior to processing the next imposed sheet. If cut and stack imposition is selected by specifying **Layout/Position/@StackOrd**, this distribution order shall be modified so that pages are distributed in the order of the individual stacks. Each stack is filled to the calculated value of **Layout/Position/@StackDepth** prior to filling the stack with the next highest value of **@StackOrd**.

Example 5.4: Cut and Stack Imposition

This simple example is configured for 2 stacks with a depth of 10 sheets. Therefore the first 20 pages will be filled into the position with **@StackOrd="0"**, the next 20 pages will be filled into the position with **@StackOrd="1"**, and the following pages will be continue switching stacks every 20 pages.

```

<Layout Automated="true" WorkStyle="WorkAndTurn">
  <Position RelativeBox="0 0 50 100" StackDepth="10" StackOrd="0"/>
  <Position RelativeBox="50 0 100 100" StackDepth="10" StackOrd="1"/>
</Layout>

```

5.4.8.3 Imposition for Tiling

Sometimes content from a surface needs to be imaged onto media that is smaller than the designated surface. Each tile SHALL be specified as a **Layout Resource** with a **Part/@TileID**. **PlacedObject/@ClipBox** SHOULD be specified as the size of the tile. **PlacedObject/@CTM** will typically be the same except for an image shift that moves the source image into the clip box.

```
<ResourceSet Name="Layout" Usage="Input">
  <Resource>
    <Part Side="Front" TileID="0 0"/>
    <Layout SurfaceContentsBox="0 0 600 420">
      <PlacedObject CTM="1 0 0 1 0 0" ClipBox="0 0 600 420" Ord="0">
        <ContentObject/>
      </PlacedObject>
    </Layout>
  </Resource>
  <!--More tiles here-->
  <Resource>
    <Part Side="Front" TileID="1 1"/>
    <Layout SurfaceContentsBox="0 0 600 420">
      <PlacedObject CTM="1 0 0 1 -600 -420" ClipBox="0 0 600 420" Ord="0">
        <ContentObject/>
      </PlacedObject>
    </Layout>
  </Resource>
</ResourceSet>
```

5.4.9 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 5.28: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.29: InkZoneCalculation – Output Resources

NAME	DESCRIPTION
InkZoneProfile	InkZoneProfile contains information about ink coverage along and perpendicular to the ink zones for a specific press geometry.

5.4.10 Interpreting

The **Interpreting** process consumes PDL data and translates a stream of display list data in a system-specified format based on information about the marking engine and media.

See **PDLCreation** for the inverse process, which consumes display list data and generates PDL.

See **RasterReading** for the process that generates display list data from raster byte map images.

See **Rendering** for the process that consumes display list data and generates raster byte map images.

Table 5.30: 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 that SHALL be interpreted.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.31: Interpreting – Output Resources

NAME	DESCRIPTION
RunList	Pipe of streamed data that represents the results of Interpreting the pages in the RunList . In general, it is assumed that the Interpreting and Rendering processes are tightly coupled and that there is no value in attempting to develop a general specification for the format of this data.

5.4.11 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 In-Design®, 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 5.32: LayoutElementProduction – Input Resources

NAME	DESCRIPTION
LayoutElementProductionParams ?	The parameters for the LayoutElementProduction process.
PreflightParams ?	Preflight profile that describes the rules that the completed RunList SHALL adhere to.
RunList ?	Location or metadata about the PDL or application file, bitmap image file, text file, vector graphics file, etc.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.33: LayoutElementProduction – Output Resources

NAME	DESCRIPTION
RunList	A RunList that represents the page elements SHALL be produced.

5.4.12 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 5.34: LayoutShifting – Input Resources

NAME	DESCRIPTION
LayoutShift	Parameters for the LayoutShifting .
RunList	References the input objects/flats to apply shifting to.
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.35: LayoutShifting – Output Resources

NAME	DESCRIPTION
RunList	The output RunList references the image data that the separation dependent layout shifts applied to.

5.4.13 PDLCreation

The [PDLCreation](#) device consumes the display list of graphical elements generated by an [Interpreting](#) or [RasterReading](#) and produces a new PDL output [RunList](#) based on the selected output parameters.

See [Interpreting](#) for the inverse process, which consumes PDL data and generates display list data.

See [RasterReading](#) for the process that generates display list data from raster byte map images.

See [Rendering](#) for the process that consumes display list data and generates raster byte map images.

Table 5.36: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.37: 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 SHALL be inserted.

5.4.14 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 page identified within the associated [RunList](#).

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 5.38: Preflight – Input Resources (Sheet 1 of 2)

NAME	DESCRIPTION
PreflightParams	A specified list of tests against which documents and/or pages are to be tested.

Table 5.38: Preflight – Input Resources (Sheet 2 of 2)

NAME	DESCRIPTION
<i>RunList</i>	The list of documents and/or pages to be preflighted.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.39: Preflight – Output Resources

NAME	DESCRIPTION
<i>PreflightReport</i>	<i>PreflightReport</i> is a container for logging information that is generated by the Preflight process.
<i>RunList ?</i>	The list of output documents that MAY have been repaired by the Preflight process.

5.4.15 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 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:

$$\text{MediaBox-width} / 72 * \text{x-resolution} = \text{width} \pm 1$$

$$\text{MediaBox-height} / 72 * \text{y-resolution} = \text{height} \pm 1$$

A gray value of 0 represents full ink, while a value of 255 represents no ink (see the DeviceGray color model in ▶ [PostScript] Chapter 4.8.2).

5.4.15.1 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%.

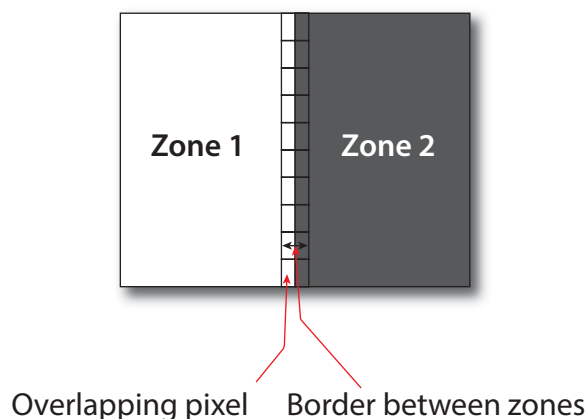
5.4.15.2 Spatial Resolution

Where pixels of the preview image 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:

$$\text{error}[\%] = \frac{100}{4 \times \text{resolution}[\text{L/mm}] \times \text{zone_width}[\text{mm}]}$$

For a 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 5-1: Worst case scenario for area coverage calculation



5.4.15.3 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:

$$\text{error}[\%] = \frac{100}{2 \times \text{number_of_shades}}$$

Therefore, at least 64 shades SHOULD be used.

5.4.15.4 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.

5.4.15.5 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).

5.4.15.6 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 5.40: PreviewGeneration – Input Resources

NAME	DESCRIPTION
ColorantControl ?	The ColorantControl resources that define the ordering and usage of inks in print modules. Needed for generating thumbnails.
PreviewGenerationParams	Parameters specifying the size and the type of the preview.
RunList ?	Bitmap data are consumed by the PreviewGeneration process.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.41: PreviewGeneration – Output Resources

NAME	DESCRIPTION
Preview	Representation of the Preview files that were produced by the PreviewGeneration process.

5.4.16 RasterReading

The [RasterReading](#) process consumes raster graphic formatted files and converts them into a display list structure.

See [Rendering](#) for the inverse process that consumes display list data and generates byte maps of raster images.

See [Interpreting](#) for the process that consumes PDL data and generates display list data.

See [PDLCreation](#) for the process that consumes display list data and generates PDL.

Table 5.42: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.43: 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 .

5.4.17 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 and with the [RenderingParams](#) information to produce binary rasterized data suitable for processes that consume [ByteMap](#) information.

See [RasterReading](#) for the inverse process that consumes raster data and generates display lists.

See [Interpreting](#) for the process that consumes PDL data and generates display list data.

See [PDLCreation](#) for the process that consumes display list data and generates PDL.

Table 5.44: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.45: Rendering – Output Resources

NAME	DESCRIPTION
RunList	Pipe of streamed data that represents the results of Rendering . This RunList MAY be consumed by any subsequent 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.

5.4.18 Screening

This process specifies the process of halftone screening. It consumes contone raster data (e.g., the output from a **RasterReading** or **Rendering** process). It produces monochrome that has been filtered through a halftone screen to identify which pixels are needed for approximating the original shades of color in the document.

This process definition includes capabilities for halftoning after raster image processing according to the PostScript definitions. Alternatively, it allows for the selection of FM screening/error diffusion techniques.

Table 5.46: Screening – Input Resources

NAME	DESCRIPTION
RunList	Ordered list of rasterized ByteMap or interpreted data representing pages or surfaces.
ScreeningParams	Parameters specifying which halftone mechanism SHALL be applied and with what specific controls.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.47: Screening – Output Resources

NAME	DESCRIPTION
RunList	Ordered list of rasterized and screened output pages. The resolution SHALL remain the same, with resulting data being one bit per component. Furthermore, the organization of planes within the data SHALL not change.

5.4.19 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 5.48: 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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.49: Separation – Output Resources

NAME	DESCRIPTION
RunList	List of separated elements, separated surfaces, separated pages or separated raster bytemaps.

5.4.20 ShapeDefProduction

This process describes the structural design of a one-up product (e.g., a non rectangular label, a box, a display, a bag, a pouch, etc.). This is a description of the unprinted blank box as it will be available after **ShapeCutting** and before **BoxFolding**. Also, this process typically (but not exclusively) describes the process of designing the shape of a new box using a CAD application. See **DieLayoutProduction** for the process of designing a die for multiple one-up products. 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: **ShapeDefProduction** needs information stored in both **ShapeDefProductionParams** and **ShapeDef** to make a new structural design.

Table 5.50: 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.
ShapeDefProductionParameters	Parameters for the structural design.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.51: ShapeDefProduction – Output Resources

NAME	DESCRIPTION
ShapeDef	A resource describing the shape of the product to be produced. If the product consists of multiple parts, the ShapeDef SHALL be partitioned by Part / @Option .

5.4.21 SheetOptimizing

[SheetOptimizing](#) describes ganging of multiple [BinderySignatures](#) onto one or more printed sheets. These [BinderySignatures](#) 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 5.52: SheetOptimizing – Input Resources

NAME	DESCRIPTION
Assembly ?	Input Assembly that MAY be used to specify the binding order e.g. for creep calculation. This Assembly MAY contain sections that are not included in this sheet optimization (e.g., when only covers are optimized and the bodies are produced individually). If the assemblies vary based on GangElement / @GangElementID , Assembly SHALL be partitioned by Part / @ProductPart and GangElement / @ExternalID SHALL match Part / @ProductPart .
BinderySignature ?	List of BinderySignature elements that describe the individual gang candidate signatures. If more than one BinderySignature resource is provided, then BinderySignature SHALL at least be partitioned by @BinderySignatureID .
SheetOptimizingParams ?	Parameters specifying details that allow individual sections to be distributed on the printed sheets.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.53: 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.

5.4.22 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](#) re-

source for the **Imposition** process. Note that the **Stripping** process can generate all resources needed for the **Imposition** process, thus also the **RunList (Marks)**.

5.4.22.1 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 **BinderySignature/@BinderySignatureType** that determines whether the pagination SHALL be explicitly defined in **SignatureCell/@FrontPages** and **SignatureCell/@BackPages**, or SHALL be calculated from **@FoldCatalog** and **@BindingOrientation** using the methods defined in ▶ Appendix F Pagination Catalog.

5.4.22.1.1 Pagination and page orientation for BinderySignatureType Fold

If **@BinderySignatureType** = "Fold", the distribution of the selected pages on a **BinderySignature** is determined by the two attributes: **@FoldCatalog** and **@BindingOrientation**. The default orientation assumes the ▶ Binding Side on the left and the ▶ Jog Edge at the bottom.

If the value of **@BindingOrientation** is one of the flip values ("Flip0", "Flip90" etc.), then the implied page ordering of the **BinderySignature** SHALL be reversed.

If the value of **@BindingOrientation** results in a binding side on the left or right, ("Rotate0" or "Rotate180") then the default alignment of page cells along the binding side SHALL be parallel.

If **@BindingOrientation** results in a binding side on the bottom or top ("Rotate90" or "Rotate270"), then the default alignment of page cells along the binding side SHALL be head to foot.

Note: This results in the default behavior that all pages are right side up when the folded **BinderySignature** is opened along the bind.

If multiple **BinderySignatures** are gathered, the flow of pages SHALL be modified by the value of **Assembly/@Order** and, if specified, **AssemblySection/@BinderySignatureID**.

If **BinderySignatures** are gathered, each **BinderySignature** consumes pages from the current front position in the document and the current position is incremented by the number of consumed pages.

If **BinderySignatures** are collected, each **BinderySignature** consumes the first half of pages from the current front position in the document and the second half of pages in reverse order from the current back position in the document. The current front position is incremented by the number of pages that were consumed from the front and current back position is decremented by the number of pages that were consumed from the back.

See also ▶ Appendix F Pagination Catalog for additional details.

Table 5.54: Stripping – Input Resources

NAME	DESCRIPTION
Assembly ?	Assembly describes how the individual BinderySignature elements are combined relative to one another to create a final product. If multiple final products are ganged on a sheet Assembly SHALL be partitioned by Part/@ProductPart .
BinderySignature ?	List of BinderySignature elements that describe the individual signatures that are combined to produce a final product. If more than one BinderySignature resource is provided, then BinderySignature SHALL at least be partitioned by @BinderySignatureID .
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. If XJDF/@Types does not contain "Imposition", then ResourceSet/@Usage of Layout SHOULD NOT be provided. Note: The previous restriction enforces that only one ResourceSet[@Name="Layout"] is provided as both input and output and will be modified appropriately.
RunList (Document) ?	List of document pages that SHALL be used to calculate the exact geometry of the Layout based on the page geometry of the pages referenced by this Layout .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.55: Stripping – Output Resources

NAME	DESCRIPTION
Layout	A Layout that describes the exact positions of the pages and SHALL be filled by the device.
RunList (Marks) ?	List of marks that SHALL be used as input of the following Imposition process.

Example 5.5: Stripping: Simple Digital Print

The following example defines a simplex layout where each surface is exactly one page.

```
<Layout Automated="true" WorkStyle="Simplex">
  <Position/>
</Layout>
```

Example 5.6: Stripping: Simple Example

This simple example specifies three 16 page bindery signatures using folding catalog scheme F16-6.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="Layout"
  JobPartID="3F-16" Types="Stripping">
  <ResourceSet Name="BinderySignature" Usage="Input">
    <Resource>
      <Part BinderySignatureID="bs1"/>
      <Part BinderySignatureID="bs2"/>
      <Part BinderySignatureID="bs3"/>
      <BinderySignature BinderySignatureType="Fold" FoldCatalog="F16-6"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Layout" Usage="Input">
    <Resource>
      <Part SheetName="sheet1"/>
      <Layout WorkStyle="WorkAndBack">
        <Position BinderySignatureID="bs1"/>
      </Layout>
    </Resource>
    <Resource>
      <Part SheetName="sheet2"/>
      <Layout WorkStyle="WorkAndBack">
        <Position BinderySignatureID="bs2"/>
      </Layout>
    </Resource>
    <Resource>
      <Part SheetName="sheet3"/>
      <Layout WorkStyle="WorkAndBack">
        <Position BinderySignatureID="bs3"/>
      </Layout>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Assembly" Usage="Input">
    <Resource>
      <Assembly BinderySignatureIDs="bs1 bs2 bs3" Order="Collecting"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

5.4.23 Trapping

The **Trapping** process modifies a set of document pages to reduce or (ideally) eliminate visible mis-registration errors in the final printed output. **XJDF** makes no assumptions about the **RunList** data. Thus **Trapping** MAY occur on PDL data, display list data or raster image data.

Table 5.56: Trapping – 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.
RunList	Structured list of incoming page contents that are to be trapped.
TrappingParams	Describes the general settings needed to perform trapping.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.57: Trapping – Output Resources

NAME	DESCRIPTION
RunList	Structured list of the modified page contents after Trapping has been executed.

5.5 Press Processes

Press processes involve the transfer of colorant to a substrate. All of the various printing technologies belong to one of two categories:

- 1 **ConventionalPrinting**, which involves printing from a physical master,
- 2 **DigitalPrinting**, which involves printing from a digital master.

The **ConventionalPrinting** and **DigitalPrinting** processes can be applied to either web or sheet fed printing.

5.5.1 ConventionalPrinting

ConventionalPrinting describes any printing process that involves printing from a physical master, including offset lithography, gravure, potato, screen and flexo printing. 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 press proof can be performed by employing a **ConventionalPrinting** process to create a **Component** with `@ProductType="Proof"`.

In the context of web printing, the **ConventionalPrinting** process SHALL be in a combined process with the **WebInlineFinishing** process.

Table 5.58: ConventionalPrinting – Input Resources (Sheet 1 of 2)

NAME	DESCRIPTION
ColorantControl ?	ColorantControl SHALL specify the complete set of colors that SHALL be printed on a sheet. The ordering of separations SHALL be specified by ColorantControl / <code>@ColorantOrder</code> .
Component	Component SHALL specify the substrate that will be printed on. The most common Component used is unprinted paper. Resource / <code>@ExternalID</code> of a Component that describes the unprinted media SHOULD be identical to the Resource / <code>@ExternalID</code> of the referenced Media .
ConventionalPrintingParams	Specific parameters to set up the press. Any process coordinate transformations that apply to ConventionalPrinting SHALL be specified in the respective parent Resource / <code>@Orientation</code> or Resource / <code>@Transformation</code> .
ExposedMedia (Plate) ?	This ExposedMedia SHALL specify the set of physical masters such as offset plates, flexo plates, screens or potatoes that SHALL be used by the ConventionalPrinting process. This ExposedMedia SHALL be partitioned by at least "Separation".

Table 5.58: ConventionalPrinting – Input Resources (Sheet 2 of 2)

NAME	DESCRIPTION
ExposedMedia (Sleeve) ?	This ExposedMedia SHALL specify the flexo sleeve if this ConventionalPrinting process describes a flexo print process.
Ink ?	Information about the physical properties of the ink. Ink SHALL be partitioned by at least " Separation ". Note: See also Color for a description of the logical properties of the color separations.
InkZoneProfile ?	The InkZoneProfile contains information about the amount ink that is needed along the printing cylinder of offset lithographic presses with ink key adjustment functions.
Layout ?	Layout MAY be used to provide the positioning of MarkObject elements such as CIELABMeasuringField , DensityMeasuringField , RegisterMark or ColorControlStrip that can be used for quality control at the press.
MiscConsumable (MountingTape) ?	Description of a mounting tape for a sleeve.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.59: ConventionalPrinting – Output Resources

NAME	DESCRIPTION
Component	Component SHALL specify the printed output. The number of copies produced SHALL be specified in <code>../Resource/AmountPool/PartAmount/@Amount</code> .

5.5.2 DigitalPrinting

DigitalPrinting is a direct printing process that, like **ConventionalPrinting**, occurs after prepress processes but before post-press 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 for each output is generated using the digital data. Electrophotography, inkjet, and other technologies are used for transferring colorant (either liquid ink or 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 5.3.1 Approval).

DigitalPrinting MAY also be 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.

Digital printing devices that 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 that includes **DigitalPrinting**. Such a combined process MAY also include other processes (e.g., **Approval**, **ColorCorrection**, **ColorSpaceConversion**, **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 that 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: Putting a label on a product or **Dropltem** is not **DigitalPrinting**; it is **Inserting**.

Table 5.60: 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	Component SHALL specify the substrate that will be printed on. The most common Component used is unprinted paper. Resource/@ExternalID of a Component that describes the unprinted media SHOULD be identical to the Resource/@ExternalID of the referenced Media .
DigitalPrintingParams	Specific parameters to set up the machinery. Any process coordinate transformations that apply to DigitalPrinting SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .

Table 5.60: DigitalPrinting – Input Resources (Sheet 2 of 2)

NAME	DESCRIPTION
<i>Ink</i> ?	Ink or toner and information that is needed for DigitalPrinting .
<i>Layout</i> ?	<i>Layout</i> MAY be used to provide the positioning of <i>MarkObject</i> elements such as <i>CIELABMeasuringField</i> , <i>DensityMeasuringField</i> , <i>RegisterMark</i> or <i>ColorControlStrip</i> that can be used for quality control at the press.
<i>RunList</i>	Raster data that will be printed on the digital press are needed for DigitalPrinting .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.61: DigitalPrinting – Output Resources

NAME	DESCRIPTION
<i>Component</i>	Components are produced for other printing processes or postpress processes. <i>Resource/AmountPool/PartAmount/@Amount</i> SHALL specify the number of copies that SHALL be produced. If the input <i>RunList</i> specifies a PDL with multiple documents or sets, such as PDF/VT, the amount SHALL BE defined as the number of sets in the input <i>RunList</i> .

5.5.3 Varnishing

Varnishing is the process of varnishing a *Component*. Spot varnishing with a ripped image or a printing plate from *ExposedMedia* SHALL be described as **DigitalPrinting** or **ConventionalPrinting** with *Ink/@InkType* = "Varnish". All types of all-over (flood) varnishing or spot varnishing applied without a ripped image or a printing plate from *ExposedMedia* SHALL be described with the **Varnishing** process. Flood coatings are typically intended to be protective, they can increase water-resistance, scuff-resistance, and even food resistance in the case of restaurant menus.

Common coating types requested by customers include UV coatings (Ultra Violet cured polymers) which provide higher durability, and aqueous coatings that are viewed as greener and typically more easily recycled at end-of-life. Both types of overall coating protect the printed image as well as the substrate.

Table 5.62: Varnishing – Input Resources

NAME	DESCRIPTION
<i>Component</i>	The <i>Component</i> to be varnished.
<i>ExposedMedia</i> ?	Various types of <i>ExposedMedia</i> MAY be specified for varnishing. See <i>VarnishingParams/@VarnishMethod</i> for details.
<i>Ink</i> ?	Details of the colorant that is used for Varnishing . <i>Ink/@InkType</i> SHOULD be "Varnish".
<i>VarnishingParams</i> ?	Details of the setup of the varnishing device. Any process coordinate transformations that apply to Varnishing SHALL be specified in the respective parent <i>Resource/@Orientation</i> or <i>Resource/@Transformation</i> .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.63: Varnishing – Output Resources

NAME	DESCRIPTION
<i>Component</i>	The varnished <i>Component</i> .

5.6 Postpress Processes

Postpress is the most flexible and varied area that is covered by this specification. The individual postpress processes are provided in alphabetical order.

5.6.1 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), head band (material and colors), kraft paper (material and glue) and tightbacking (different geometry and measurements).

Table 5.64: 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. Any process coordinate transformations that apply to BlockPreparation SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
MiscConsumable (RegisterRibbon) ?	Description of the register ribbons. If present, processing instructions such as ribbon lengths should be specified in BlockPreparationParams/RegisterRibbon .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.65: BlockPreparation – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the prepared book block. The value of Component/@ProductType SHALL be "BookBlock".

5.6.2 BoxFolding

BoxFolding defines the process of folding and gluing blanks into folded flat boxes for packaging.

Table 5.66: 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 @ProductType = "BlankBox".
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.67: BoxFolding – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the folded flat box. The value of Component/@ProductType SHALL be "FlatBox".

5.6.3 BoxPacking

A pile, stack or bundle of products can be packed into a box or carton.

Table 5.68: BoxPacking – Input Resources (Sheet 1 of 2)

NAME	DESCRIPTION
BoxPackingParams	Specific parameters to set up the machinery.
Bundle ?	Bundle describes the structure of the packed boxes that are represented by the output Component .

Table 5.68: BoxPacking – Input Resources (Sheet 2 of 2)

NAME	DESCRIPTION
Component (Contents) ?	The BoxPacking process puts a set of Product or Component resources into the Component(Box) . If more than one Component(Contents) resource is specified, Bundle/BundleItem/@ItemRef SHALL also be specified for each 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.
MiscConsumable (FillMaterial) ?	Additional details of the filler material. MiscConsumable/@Type SHOULD be one of "BlisterPack", "Paper" or "Styrofoam".
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.69: BoxPacking – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the Component that represents the packed Box.

5.6.4 Bundling

The **Bundling** process is normally followed by a **Strapping** process. In a **Bundling** process, single products like sheets or signatures are bundled together. The resulting bundle is the output **Component** of the process and is used to store the products. When this **Component** is used as an input to a consuming or subsequent process (e.g., **Gathering**, **Collecting** or **Inserting**), the single components of a bundle are used.

Figure 5-2: Bundle creation



Figure 5-3: Bundle transport



Table 5.70: Bundling – Input Resources (Sheet 1 of 2)

NAME	DESCRIPTION
Bundle ?	Bundle describes the structure of the input Component that SHALL be bundled.

Table 5.70: Bundling – Input Resources (Sheet 2 of 2)

NAME	DESCRIPTION
<i>BundlingParams</i>	Bundling parameters.
<i>Component</i>	The <i>Component</i> to be bundled.
<i>Media</i> ?	End boards to protect the bundle. For each bundle a pair of end boards is needed.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.71: Bundling – Output Resources

NAME	DESCRIPTION
<i>Component</i>	The completed bundle.

5.6.5 CaseMaking

Case making is the process where a hardcover book case is produced.

Table 5.72: CaseMaking – Input Resources

NAME	DESCRIPTION
<i>CaseMakingParams</i>	Specific parameters to set up the machinery. Any process coordinate transformations that apply to <i>CaseMaking</i> SHALL be specified in the respective parent <i>Resource</i> / <i>@Orientation</i> or <i>Resource</i> / <i>@Transformation</i> .
<i>Component</i> (<i>CoverMaterial</i>) ?	The cover material is either a preprinted or processed sheet of paper.
<i>Component</i> (<i>CoverBoard</i>)	The cardboard <i>Component</i> used for the cover board.
<i>Component</i> (<i>SpineBoard</i>) ?	The cardboard <i>Component</i> used for the spine board. If not specified, the <i>Component</i> (<i>CoverBoard</i>) SHALL be used for the spine board.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.73: CaseMaking – Output Resources

NAME	DESCRIPTION
<i>Component</i>	One <i>Component</i> is produced: the book case. <i>Component</i> / <i>@ProductType</i> SHALL be "BookCase".

5.6.6 CasingIn

The hardcover book case and the book block are joined in the *CasingIn* process.

Table 5.74: CasingIn – Input Resources

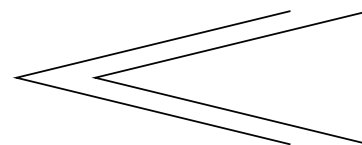
NAME	DESCRIPTION
<i>CasingInParams</i>	Specific parameters to set up the machinery.
<i>Component</i> (<i>BookBlock</i>)	The prepared book block.
<i>Component</i> (<i>BookCase</i>)	The hardcover book case.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.75: CasingIn – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the completed hardcover book.

5.6.7 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 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 5.76: Collecting – Input Resources

NAME	DESCRIPTION
Assembly ?	Assembly explicitly describes the sequence of the Component resources to be collected. If Assembly is not specified, the sequence SHALL be defined by the sequence of the Component .
Component	The Component Resource elements in the ResourceSet represent the individual signatures that shall be collected. The first Resource element in XML order shall represent the outer Component .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.77: Collecting – Output Resources

NAME	DESCRIPTION
Component	A block of collected sheets is produced. This Component can be joined in further post-press processes.

5.6.8 CoverApplication

CoverApplication describes the process of applying a softcover to a book block.

Table 5.78: CoverApplication – Input Resources

NAME	DESCRIPTION
Component	This Component ResourceSet SHALL represent the cover and book block. Exactly one Component resource with Component/@ProductType = "Cover" SHALL be specified. The other Component resources SHALL represent the book block or parts of the book block.
CoverApplicationParams	Specific parameters to set up the machinery.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.79: CoverApplication – Output Resources

NAME	DESCRIPTION
Component	The book block with the applied softcover.

5.6.9 Creasing

Sheets are creased or grooved to enable folding or to create even, finished page delimiters.

Table 5.80: Creasing – Input Resources

NAME	DESCRIPTION
Component	This process consumes one Component .
CreasingParams	Details of the Creasing process. Any process coordinate transformations that apply to Creasing SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.81: Creasing – Output Resources

NAME	DESCRIPTION
Component	One creased Component is produced.

5.6.10 Cutting

Sheets are cut using a guillotine **Cutting** machine.

Since **Cutting** is described here in the most machine independent manner, 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 5.82: Cutting – Input Resources

NAME	DESCRIPTION
Component	This process consumes one Component : the printed sheets.
CuttingParams	Details of the Cutting process. Any process coordinate transformations that apply to Cutting SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.83: Cutting – Output Resources

NAME	DESCRIPTION
Component	One or several blocks of cut Component resources are produced. The output SHOULD be partitioned by CutBlock .

5.6.11 DieMaking

This process describes the production of tools for a die cutter (e.g., in a die maker shop).

Table 5.84: DieMaking – Input Resources

NAME	DESCRIPTION
DieLayout	A resource describing the die cutter tool set.
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.85: DieMaking – Output Resources

NAME	DESCRIPTION
Tool	The set of tools for the die cutter. If the tool set contains multiple parts, e.g. an upper and a lower die, the Tool SHALL be partitioned by Part/@Option and/or Part/@Side .

5.6.12 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 5.86: Embossing – Input Resources

NAME	DESCRIPTION
Component	This process consumes one Component that is embossed by the process.
EmbossingParams	Parameters to set up the machinery. Any process coordinate transformations that apply to Embossing SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Media(Foil) ?	Media(Foil) SHOULD be provided if an EmbossingParams/Emboss/@EmbossingType="FoilEmbossing" or "FoilStamping" .
Tool ?	The embossing stamps or calenders. If the tool set contains multiple parts, e.g. an upper and a lower stamp, the Tool SHALL be partitioned by Part/@Option and/or Part/@Side .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.87: Embossing – Output Resources

NAME	DESCRIPTION
Component	One Component is created.

5.6.13 EndSheetGluing

EndSheetGluing finalizes the book block in preparation for case binding by attaching end sheets to the book block. 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 5.88: EndSheetGluing – Input Resources

NAME	DESCRIPTION
Component	A back end sheet and a front end sheet are glued onto the book block.
Component (BackEndSheet) ?	A back end sheet that SHALL be mounted on the book block. At least one of Component (BackEndSheet) or Component (FrontEndSheet) SHALL be present.
Component (FrontEndSheet) ?	A front end sheet that SHALL be mounted on the book block. At least one of Component (BackEndSheet) or Component (FrontEndSheet) SHALL be present.
EndSheetGluingParams	Specific parameters to set up the machinery.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.89: EndSheetGluing – Output Resources

NAME	DESCRIPTION
Component	A book block is produced that includes the end sheets.

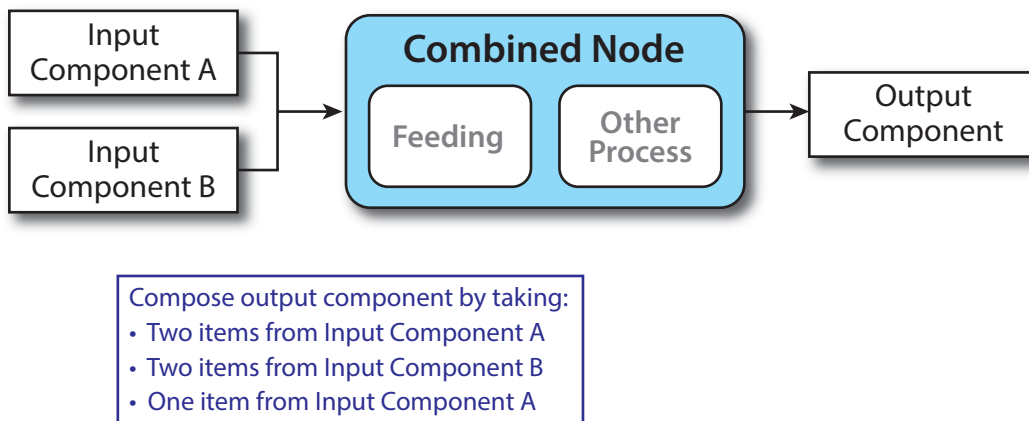
5.6.14 Feeding

The **Feeding** process separates sheets or signatures from a stack, roll or stream and feeds single **Components** to processes such as **Folding**, **Gathering**, **Collecting**, **ConventionalPrinting**, etc.

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 **Components** from a roll.

Figure 5-4: Combined process with Feeding process



In our example above, one input component (Component A) consists 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 5-5: Input components. ▶ Figure 5-6: Output component shows the output for the case of **Gathering**.

Figure 5-5: Input components

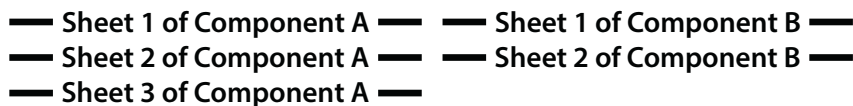
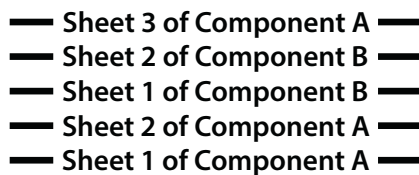


Figure 5-6: Output component



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 5.90: Feeding – Input Resources

NAME	DESCRIPTION
Component +	Sheets or signatures to be fed to the machinery. ResourceSet/@ProcessUsage of a Component MAY be specified as any valid @ProcessUsage of the feed consuming process.
FeedingParams	Specific parameters to set up the Feeding process.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.91: Feeding – Output Resources

NAME	DESCRIPTION
Component +	Component (s) fed to the consuming process.

5.6.15 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. 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 when 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 **FoldingParams/Crease** and **FoldingParams/Perforate** respectively.

Table 5.92: 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. Any process coordinate transformations that apply to Folding SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

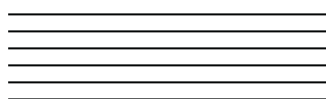
Table 5.93: Folding – Output Resources

NAME	DESCRIPTION
Component	The process produces a Component , which in most cases is a folded sheet.

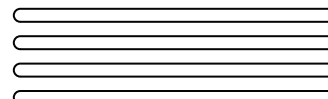
5.6.16 Gathering

In the **Gathering** process, sheets or other **Component** resources are accumulated on a pile. ▶ Figure 5-7: Gathering shows typical gathered piles.

Figure 5-7: Gathering



Loose sheets



Folded signatures

Table 5.94: 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 .
Component	The Component Resource elements in the ResourceSet represent the individual signatures that SHALL be gathered on a pile. The first Resource element in XML order SHALL represent the top Component .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.95: Gathering – Output Resources

NAME	DESCRIPTION
Component	Components gathered together (e.g., a pile of folded sheets).

5.6.17 Gluing

Gluing describes arbitrary methods of applying glue to a **Component**.

Table 5.96: Gluing – Input Resources

NAME	DESCRIPTION
Component	This process consumes one Component : the printed sheets.
GluingParams	Details of the Gluing process. Any process coordinate transformations that apply to Gluing SHALL be specified in the respective parent Resource / @Orientation or Resource / @Transformation .
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.97: Gluing – Output Resources

NAME	DESCRIPTION
Component	One Component is produced, the input Component with glue applied to it.

5.6.18 HeadBandApplication

Head bands are applied to the hardcover book block. In case different head bands are desired for top and bottom, **MiscConsumable**(**Headband**) SHOULD be partitioned with **Part**/**@Option** = "Top" and **Part**/**@Option** = "Bottom".

Table 5.98: HeadBandApplication – Input Resources

NAME	DESCRIPTION
Component	The prepared book block.
HeadBandApplicationParams	Specific parameters to set up the machinery.
MiscConsumable (BackReinforcement) ?	Additional details such as color and brand of the back reinforcement MAY be specified in this MiscConsumable . If required, the strip material SHOULD be specified in MiscConsumable / @TypeDetails .
MiscConsumable (Headband) ?	Additional details such as color and brand of the head band MAY be specified in this MiscConsumable .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.99: HeadBandApplication – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the hardcover block with head bands.

5.6.19 HoleMaking

A variety of machines (e.g., those responsible for stamping and drilling) can perform the **HoleMaking** process.

HoleMaking MAY be used to describe line hole punching that 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 is typically 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, **HoleMakingParams/HolePattern/@Center** applies to the initial hole and **HoleMakingParams/HolePattern/@Extent** applies to each hole individually.

Sometimes line hole punching is performed for multiple webs before dividing the web after the **HoleMaking** process.

The following figure shows the parameters for both cases.

Figure 5-8: HolePattern parameters

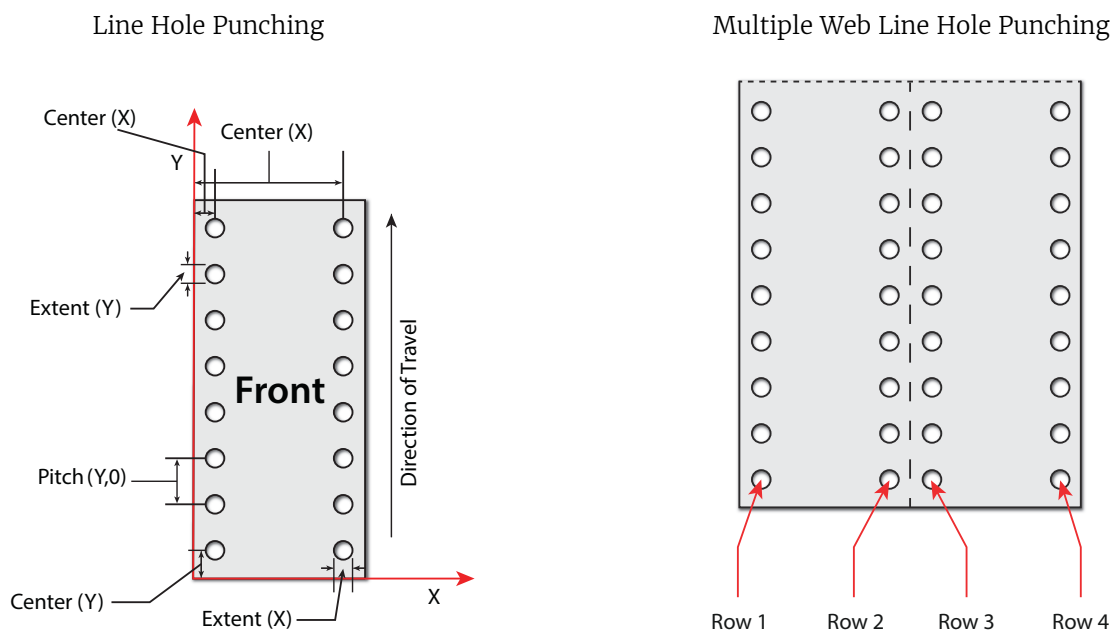


Table 5.100: HoleMaking – Input Resources

NAME	DESCRIPTION
Component	One Component (e.g., a printed sheet or a pile of sheets) is modified in the HoleMaking process.
HoleMakingParams	Specific parameters, including hole diameter and positions, used to set up the machinery. Any process coordinate transformations that apply to HoleMaking SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

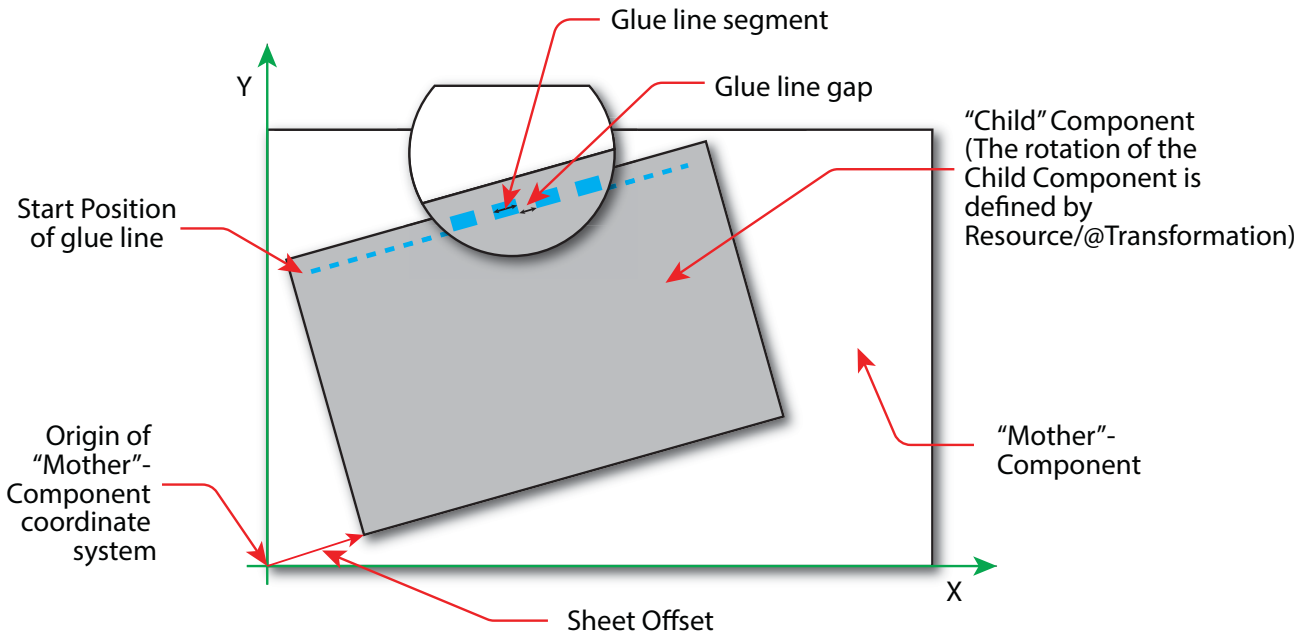
Table 5.101: HoleMaking – Output Resources

NAME	DESCRIPTION
Component	A Component with holes (e.g., a book block or a single sheet) is produced.

5.6.20 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 multiple child components is specified as a combined process with multiple individual **Inserting** steps.

Figure 5-9: 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.

Table 5.102: Inserting – Input Resources

NAME	DESCRIPTION
Component	Designates where to insert the child Component .
Component (Child)	The Component that SHALL be inserted in the mother Component . Any coordinate transformations that apply to the child Component SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
InsertingParams	Specific parameters (e.g., placement) to set up the machinery.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.103: Inserting – Output Resources

NAME	DESCRIPTION
Component	A mother Component is produced containing the inserted child Component .

5.6.21 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 5.104: Jacketing – Input Resources

NAME	DESCRIPTION
Component (Book)	The book that the jacket is wrapped around.
Component (Jacket)	The description of the jacket.
JacketingParams	Specific parameters to set up the machinery.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.105: Jacketing – Output Resources

NAME	DESCRIPTION
Component	The jacketed book.

5.6.22 Labeling

A label can be attached to a **Component**. The label can contain information on the addressee, the product, the product quantities, etc., which can be different for each **Component**.

Table 5.106: Labeling – Input Resources

NAME	DESCRIPTION
Bundle ?	Bundle describes the structure of the input Component that SHALL be labeled.
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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.107: Labeling – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the labeled Component .

5.6.23 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 5.108: Laminating – Input Resources (Sheet 1 of 2)

NAME	DESCRIPTION
Component	A Component SHALL be specified for Laminating .
LaminatingParams	Specific parameters to set up the machinery. Any process coordinate transformations that apply to Laminating SHALL be specified in the respective parent Resource / @Orientation or Resource / @Transformation .
Media (Foil) ?	The laminating foil material.
MiscConsumable (Glue)	Details of the dispersion glue used if LaminatingParams / @LaminatingMethod ="DispersionGlue".

Table 5.108: Laminating – Input Resources (Sheet 2 of 2)

NAME	DESCRIPTION
MiscConsumable (Hardener)	Details of the dispersion glue hardener used if LaminatingParams/@LaminatingMethod="DispersionGlue" .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.109: Laminating – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the laminated component.

5.6.24 LooseBinding

LooseBinding describes a binding (as for a notebook, catalog, price list) holding pages together by spiral wire, plastic combs, metal clamps, rings or metal rods.

5.6.24.1 Loose-Leaf Binding Methods

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 ring binding method is the most prominent binding in the loose-leaf binding category. Loose-leaf binding methods include:

- Ring binding.
In this process, pre-punched 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 are used. Additional details MAY be specified in [LooseBindingParams/RingBindingDetails](#).
- 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. Mechanical binding methods include:
 - Channel binding.
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.
Additional details MAY be specified in [LooseBindingParams/ChannelBindingDetails](#).
 - Coil binding.
Another name is spiral binding. Metal wire, wire with plastic or pure plastic is used to fasten pre-punched 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.
Additional details MAY be specified in [LooseBindingParams/CoilBindingDetails](#).
 - Comb binding.
In this method, a metal wire, wire with plastic or pure plastic insert wraps through pre-punched holes in the substrate. In case of plastic combs, these holes are most often rectangular and elongated. After the comb is opened with a special tool, the pre-punched block of sheets—often together with a top and bottom cover—is inserted onto the “teeth” of the comb.
Additional details MAY be specified in [LooseBindingParams/CombBindingDetails](#).
 - Strip binding.
Hard plastic strips are held together by plastic pins, which in turn are bound to the strips with heat. The sheets to be bound are pre-punched so that the top strip with multiple pins fits through the assembled material. The top strip 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.
Additional details MAY be specified in [LooseBindingParams/StripBindingDetails](#).

Table 5.110: LooseBinding – Input Resources

NAME	DESCRIPTION
Component	This Component ResourceSet SHALL represent sheets to be bound. At most one Component resource with Component/@ProductType = "Cover" SHALL be specified. The other Component resources SHALL represent the sheets to be bound.
LooseBindingParams	Specific parameters to set up the machinery. Any process coordinate transformations that apply to LooseBinding SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
MiscConsumable (Cover) ?	Details such as brand and color of the cover. If the binding mechanism is separate from the cover, the binding mechanism SHALL be specified in MiscConsumable (Spine) . The cover is dependent on the binding type. For ring binding, where @BindingType="RingBinding" , the value of MiscConsumable/@Type SHOULD be "RingBinder" . For all other types of binding any non-printed cover, e.g. a transparent plastic sheet, the value of MiscConsumable/@Type SHOULD be "Cover" .
MiscConsumable (Spine) ?	Details such as brand and color of the spine. The spine is dependent on the binding type: ChannelBinding : The clamp/cover. The value of MiscConsumable/@Type SHOULD be "ChannelBinder" . CoilBinding : The coil. The value of MiscConsumable/@Type SHOULD be "Coil" . CombBinding : The comb. The value of MiscConsumable/@Type SHOULD be "Comb" . StripBinding : The strip. The value of MiscConsumable/@Type SHOULD be "StripBinder" .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.111: LooseBinding – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the bound item such as a brochure.

5.6.25 Palletizing

Bundles, stacks, piles or boxes can be loaded onto a pallet.

Table 5.112: Palletizing – Input Resources

NAME	DESCRIPTION
Bundle ?	Bundle describes the structure of the loaded pallets that are represented by the output Component .
Component ?	The Palletizing process puts the set of Product or Component resources onto the pallet. If more than one Component resource is specified, Bundle/BundleItem/@ItemRef SHALL also be specified for each Component .
Pallet	The pallet.
PalletizingParams	Specific parameters to set up the machinery.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.113: Palletizing – Output Resources

NAME	DESCRIPTION
Component	One Component is produced. It represents the loaded pallet.

5.6.26 Perforating

Perforating describes any process where a **Component** is perforated.

Table 5.114: Perforating – Input Resources

NAME	DESCRIPTION
Component	This process consumes one Component : the printed sheets.
PerforatingParams	Details of the Perforating process. Any process coordinate transformations that apply to Perforating SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.115: Perforating – Output Resources

NAME	DESCRIPTION
Component	One Component is produced.

5.6.27 ShapeCutting

The **ShapeCutting** process can be performed using tools such as hollow form punching, perforating or die-cutting equipment.

Table 5.116: ShapeCutting – Input Resources

NAME	DESCRIPTION
Component	This process consumes one Component that are the sheets to be cut.
ShapeCuttingParams ?	Details of the ShapeCutting process. Any process coordinate transformations that apply to ShapeCutting SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Tool ?	The set of tools (die, counter, blankers, strippers, etc.). If the tool set contains multiple parts, e.g. an upper and a lower die, the Tool SHALL be partitioned by Part/@Option and/or Part/@Side .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.117: ShapeCutting – Output Resources

NAME	DESCRIPTION
Component	The Component SHOULD be partitioned by @StationName .

5.6.28 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 5.6.39 Wrapping.

Table 5.118: Shrinking – Input Resources (Sheet 1 of 2)

NAME	DESCRIPTION
Bundle ?	Bundle describes the structure of the input Component that SHALL be treated.
Component	The Bundle including the shrink-wrap media is represented by this Component .

Table 5.118: Shrinking – Input Resources (Sheet 2 of 2)

NAME	DESCRIPTION
ShrinkingParams	Specific parameters to set up the machinery.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.119: Shrinking – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the bundle including the shrunk shrink-wrap media.

5.6.29 SpinePreparation

The [SpinePreparation](#) process describes the preparation of the spine of book blocks for hard and softcover book production (e.g., milling and notching).

Table 5.120: SpinePreparation – Input Resources

NAME	DESCRIPTION
Component	The raw book block.
SpinePreparationParams	Specific parameters to set up the machinery. Any process coordinate transformations that apply to SpinePreparation SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.121: SpinePreparation – Output Resources

NAME	DESCRIPTION
Component	The book block with a processed spine.

5.6.30 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 hardcover book block.

Table 5.122: SpineTaping – Input Resources

NAME	DESCRIPTION
Component	The book block that the spine is taped to.
MiscConsumable (Tape) ?	Details such as brand and color of the tape.
SpineTapingParams	Specific parameters to set up the machinery. Any process coordinate transformations that apply to SpineTaping SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.123: SpineTaping – Output Resources

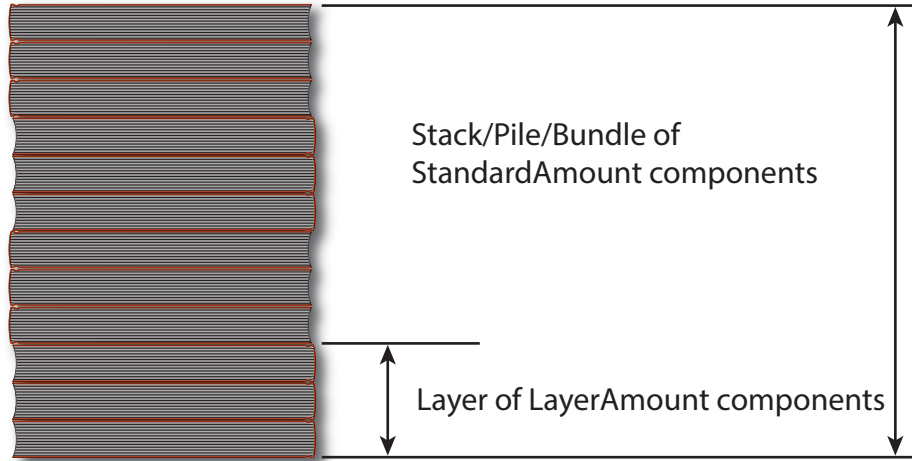
NAME	DESCRIPTION
Component	The book block with the spine.

5.6.31 Stacking

The **Stacking** process collects **Components** 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.

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 5-10: Stacking layers).

Figure 5-10: 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.

Table 5.124: Parameters in Stacking^a








PILE PATTERN	STANDARD AMOUNT	LAYER AMOUNT	COMPENSATE	DISJOINING OFFSET
	6	6	true	0 0
	6	1	true	0 0
	6	1	false	x 0
	6	1	true	x 0

Table 5.124: Parameters in Stacking^a

PILE PATTERN	STANDARD AMOUNT	LAYER AMOUNT	COMPENSATE	DISJOINTING OFFSET
	6	3	true	0 0
	6	3	false	x 0
	6	3	true	x 0

a. Column headings 'STANDARD AMOUNT', 'LAYER AMOUNT', 'COMPENSATE' and 'DISJOINTING OFFSET' refer to the values in *StackingParams/@StandardAmount*, *StackingParams/@LayerAmount*, *StackingParams/@Compensate* and *StackingParams/Disjointing/@Offset* respectively.

If the number of components is not evenly divisible by *StackingParams/@StandardAmount* or the number of components in a bundle is not evenly divisible by *StackingParams/@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. *StackingParams/@MinAmount* and *StackingParams/@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 5-11: Odd count handling for a bundle

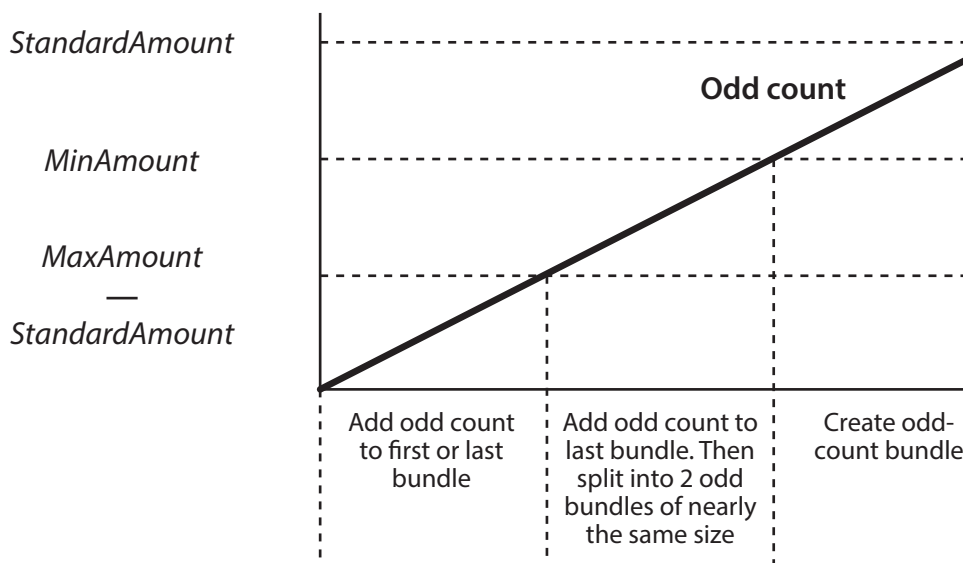


Figure 5-12: Odd count handling for a layer

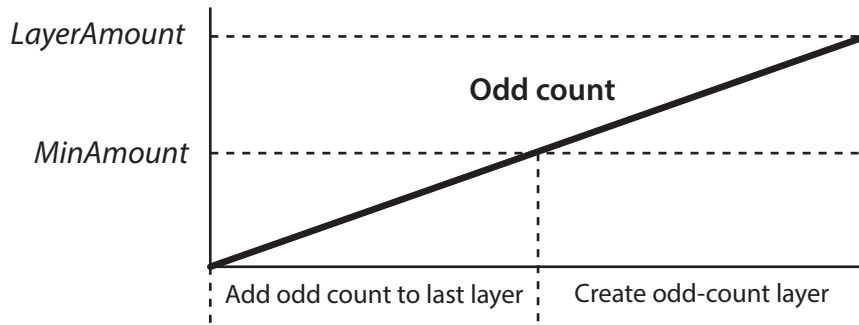


Table 5.125: Stacking – Input Resources

NAME	DESCRIPTION
<i>Bundle</i> ?	<i>Bundle</i> describes the structure of the stacks that are represented by the output <i>Component</i> .
<i>Component</i>	The <i>Stacking</i> process consumes one <i>Component</i> and stacks it onto a stack.
<i>StackingParams</i>	Specific parameters to set up the machinery. Any process coordinate transformations that apply to <i>Stacking</i> SHALL be specified in the respective parent <i>Resource</i> / <i>@Orientation</i> or <i>Resource</i> / <i>@Transformation</i> .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.126: Stacking – Output Resources

NAME	DESCRIPTION
<i>Component</i>	One <i>Component</i> is produced: the stack of input components.

5.6.32 Stitching

Gathered or collected sheets or signatures are stitched together with a cover. This process can be used to describe corner stitching, side stitching or saddle stitching.

Table 5.127: Stitching – Input Resources

NAME	DESCRIPTION
<i>Component</i>	A <i>Component</i> SHALL be specified that represents the pile of gathered or collected sheets, including the cover.
<i>MiscConsumable</i> (<i>Wire</i>) ?	Details such as brand and color of the stitching wire.
<i>StitchingParams</i>	Specific parameters to set up the machinery. Any process coordinate transformations that apply to <i>Stitching</i> SHALL be specified in the respective parent <i>Resource</i> / <i>@Orientation</i> or <i>Resource</i> / <i>@Transformation</i> .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.128: Stitching – Output Resources

NAME	DESCRIPTION
<i>Component</i>	One <i>Component</i> is produced: the gathered or collected sheets including the cover stitched together.

5.6.33 Strapping

The Strapping process specifies how straps are wrapped around a bundle. The straps that are used SHOULD be specified as a **MiscConsumable**.

Table 5.129: Strapping – Input Resources

NAME	DESCRIPTION
Bundle ?	Bundle describes the structure of the input Component that SHALL be strapped.
Component	The Strapping process puts straps around a bundle that is represented by a Component .
MiscConsumable (Strap) ?	Details such as brand and color of the strap.
StrappingParams	Specific parameters to set up the machinery. Any process coordinate transformations that apply to Strapping SHALL be specified in the respective parent Resource / @Orientation or Resource / @Transformation .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.130: Strapping – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the strapped Component .

5.6.34 ThreadSealing

ThreadSealing involves sewing the spines of individual signatures of a book with pieces of meltable thread prior to **Gathering**. The thread is melted by applying heat during **SpinePreparation**. In practice, **ThreadSealing** will often be combined with **Folding** in a single process.

Table 5.131: ThreadSealing – Input Resources

NAME	DESCRIPTION
Component	This process consumes one Component that is the gathered individual folded signatures.
MiscConsumable (Thread) ?	Details such as brand and color of the thread.
ThreadSealingParams	Details of the ThreadSealing process. Any process coordinate transformations that apply to ThreadSealing SHALL be specified in the respective parent Resource / @Orientation or Resource / @Transformation .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.132: ThreadSealing – Output Resources

NAME	DESCRIPTION
Component	One Component is produced, the individual folded and sewn signatures.

5.6.35 ThreadSewing

This process involves stitching signatures together with thread to create a book block.

Table 5.133: ThreadSewing – Input Resources (Sheet 1 of 2)

NAME	DESCRIPTION
Component	The operation requires one Component that is the gathered individual folded signatures.

Table 5.133: ThreadSewing – Input Resources (Sheet 2 of 2)

NAME	DESCRIPTION
<i>MiscConsumable</i> (Thread) ?	Details such as brand and color of the thread.
<i>ThreadSewingParams</i>	Specific parameters to set up the machinery. Any process coordinate transformations that apply to ThreadSewing SHALL be specified in the respective parent <i>Resource/</i> <i>@Orientation</i> or <i>Resource/</i> <i>@Transformation</i> .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.134: ThreadSewing – Output Resources

NAME	DESCRIPTION
<i>Component</i>	One Component is produced: the thread-sewn components forming an item such as a raw book block.

5.6.36 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 prior to a **Trimming** process.

The process coordinate system is defined as follows:

- The X-axis SHALL be aligned with the registered side. It increases from the binding side to the face side.
- The Y-axis SHALL be aligned with the binding side. It increases from the registered edge.

Figure 5-13: Parameters and coordinate system used for trimming

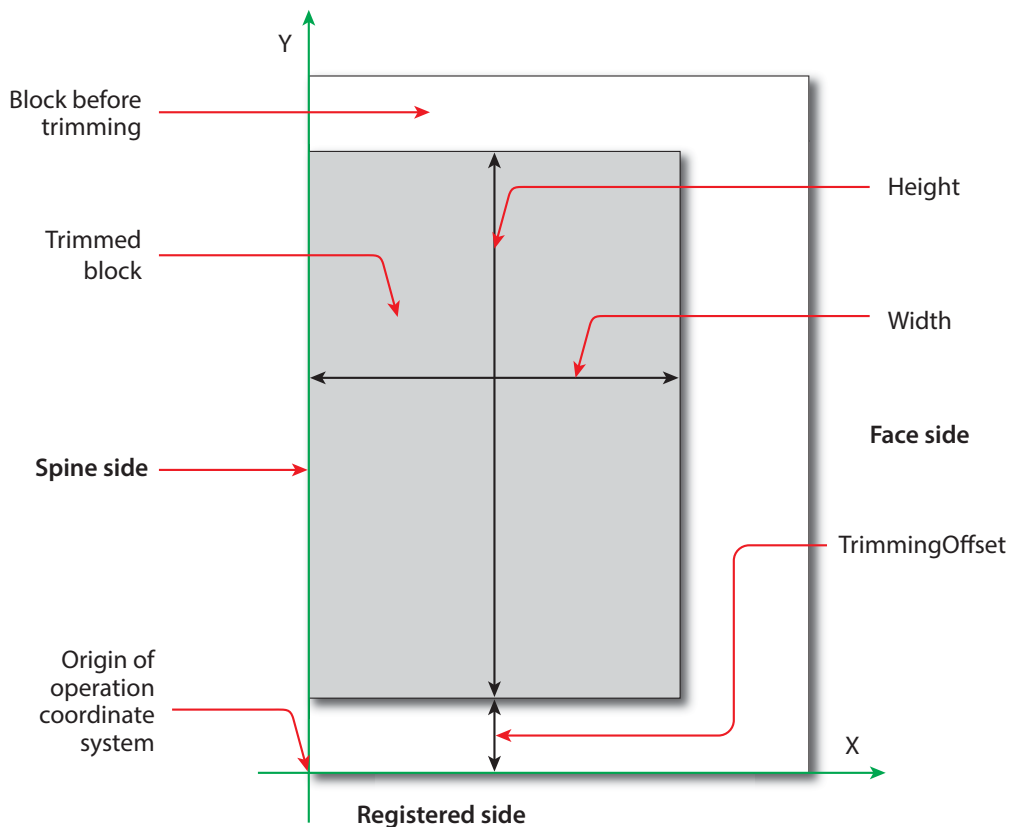


Table 5.135: Trimming – Input Resources

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. Any process coordinate transformations that apply to Trimming SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.136: Trimming – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the trimmed component.

5.6.37 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, such as **Stitching**, **Trimming**, **Gluing**, etc.

Table 5.137: WebInlineFinishing – Input Resources

NAME	DESCRIPTION
Component	Printed webs or ribbons, which will be processed by the WebInlineFinishing process.
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 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.138: 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 .

5.6.38 Winding

The **Winding** process describes the winding of continuous media or processed components onto a core or roll stand. The setup is defined in **WindingParams**. The final orientation of the labels or components on the output roll is specified in **Component/@WindingResult**.

Table 5.139: Winding – Input Resources (Sheet 1 of 2)

NAME	DESCRIPTION
Component	Ribbon or web to be wound.
Media (Core) ?	Core that the input Component is wound around.

Table 5.139: Winding – Input Resources (Sheet 2 of 2)

NAME	DESCRIPTION
WindingParams ?	Setup parameters of the winding process. Any process coordinate transformations that apply to Winding SHALL be specified in the respective parent Resource/@Orientation or Resource/@Transformation .
Generic Input Resources*	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.140: 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.

5.6.39 Wrapping

Single products, bundles or pallets can be wrapped using bags, bands or wrapping material.

Table 5.141: Wrapping – Input Resources

NAME	DESCRIPTION
Bundle ?	Bundle describes the structure of the input Component that SHALL be wrapped.
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 wrap- ping material.
MiscConsumable (Wrapper) ?	Additional details of the wrapper material. Non-printed material SHOULD be repre- sented as MiscConsumable . MiscConsumable(Wrapper) SHALL NOT be present if Component (Wrapper) is provided. MiscConsumable/@Type SHOULD be one of "PaperBand", "PaperWrap", "PlasticBand", "RubberBand" or "ShrinkWrap".
WrappingParams	Specific parameters to set up the machinery.
Generic Input Resources *	See ▶ Table 5.1 Generic Input ResourceSets for additional input resources that are valid for all process types.

Table 5.142: Wrapping – Output Resources

NAME	DESCRIPTION
Component	One Component is produced: the wrapped Component .

6 Resources

This chapter provides a list (in alphabetical order) of all specific resource types.

6.1 Resource

Resource elements are child elements of a **ResourceSet** and describe the physical or logical entity in the partition context that is defined in **Resource/Part**. For instance a **ResourceSet/@Name="ExposedMedia"** can specify a set of printing plates and each child **Resource** element will describe an individual plate.

Table 6.1: Resource Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Brand</i> ?	string	Brand or manufacturer of the Resource .
<i>CommentURL</i> ?	URL	URL to an external, human-readable description 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 that describe physical entities for communication from applications to humans in order to reference the Resource .
<i>Duration</i> ?	duration	If <i>@Duration</i> is specified for ResourceSet/@Usage="Input" , <i>@Duration</i> specifies the time duration during which the Resource will be or has been used. Note: <i>@Duration</i> in conjunction with <i>@Start</i> or <i>@StartOffset</i> can be used to schedule or track resources such as tools that are only required during part of the processing time as defined in NodeInfo . If <i>@Duration</i> is specified for ResourceSet/@Usage="Output" , <i>@Duration</i> specifies the time that the Resource SHALL be or has been stored after it has been produced. Note: <i>@Duration</i> can be used to define resting periods, e.g. to allow press sheets to dry prior to further processing. Note: If <i>@Duration</i> is specified in descendents of ResourceInfo it SHALL specify actual duration. In all other cases it SHALL specify a planned or requested duration.
<i>ExternalID</i> ?	NMTOKEN	An identifier of the resource as defined in the MIS system. For instance item codes or article numbers or identifiers on semi-finished products. <i>@ExternalID</i> SHALL be used to uniquely identify resources and products for the purpose of inventory tracking.
<i>GrossWeight</i> ?	float	Gross weight of a single Resource , as counted in <i>@Amountxxx</i> , in grams.
<i>ID</i> ?	ID	Unique XJDF internal identifier of a Resource .
<i>Orientation</i> ?	enumeration	Named orientation describing the orientation of a Resource relative to the ideal process coordinate that uses this Resource as input or output. If <i>@Orientation</i> is specified for an output Resource , the XJDF that processes the Resource SHALL 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 <i>@Orientation</i> or <i>@Transformation</i> SHALL be specified. For details on coordinate systems, see ▶ Section 2.6 Coordinate Systems in XJDF. Allowed value is from: ▶ Orientation.
<i>ResourceWeight</i> ?	float	Net weight of a single Resource , as counted in <i>@Amount</i> , in grams.

Table 6.1: Resource Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Start</i> ?	dateTime	Time and date when the usage of the Resource SHALL start or has started. If <i>@Start</i> is specified in descendants of ResourceInfo it SHALL specify an actual start time. In all other cases, it SHALL specify a planned or requested start time.
<i>StartOffset</i> ?	duration	Offset time when the Resource is scheduled to be used after processing has begun. <i>@StartOffset</i> SHALL NOT be specified if <i>@Start</i> is present. <i>@StartOffset</i> SHALL NOT be specified in the context of ResourceInfo .
<i>Status</i> ?	enumeration	The status of a resource indicates whether it is available for processing. <i>@Status</i> SHALL NOT be specified if ResourceSet / <i>@Usage</i> ="Output". Allowed values are: Available – Indicates that the resource is available for processing. Unavailable – Indicates that the resource is not available for processing.
<i>Transformation</i> ?	matrix	Matrix describing the orientation of a Resource relative to the ideal process coordinate using this Resource as input or output. If <i>@Transformation</i> is specified for an output Resource , the XJDF that processes the Resource SHALL 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 <i>@Orientation</i> or <i>@Transformation</i> SHALL be specified. For details on coordinate systems, see ▶ Section 2.6 Coordinate Systems in XJDF.
<i>AmountPool</i> ?	element	AmountPool specifies partial amounts and waste for this Resource .
<i>Comment</i> *	element	Any human-readable text that describes the Resource .
<i>GeneralID</i> *	element	Additional identifiers related to the Resource .
<i>Part</i> *	element	The Part elements identify the partition context of the Resource element. The structure of the Part element is defined in ▶ Table 6.4 Part Element. For details on partitioned Resource elements, see ▶ Section 6.1.3.2 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 entity that applies to multiple partitions (e.g., the color plates that apply to all versions of a multi version job).
<i>Specific Resource</i> ?	element	Details of the Resource . The XML element name SHALL be the value of ResourceSet / <i>@Name</i> . If the specific resource is defined in the XJDF namespace, then it SHALL have the prefix that is declared in the <i>xmlns</i> attribute of the root element. Specific resource SHALL be specified as the last XJDF namespace element in the Resource . Note: This is an exception to the general instruction that all elements are ordered alphabetically.
<foreign namespace elements> *	element	Any elements in a foreign namespace. Foreign namespace extensions SHOULD NOT duplicate functionality of XJDF . Foreign namespace extensions SHALL be specified after all elements in the XJDF namespace.

6.1.1 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**.

The interpretation of the amounts specified in an **AmountPool** depends on the context of the **AmountPool**, i.e. **AmountPool** elements that are specified in descendants of **ResourceInfo** elements SHALL specify actual amounts. All other **AmountPool** elements SHALL specify planned or requested amounts.

Table 6.2: AmountPool Element

NAME	DATA TYPE	DESCRIPTION
<i>PartAmount</i> +	element	PartAmount SHALL specify the amounts and waste of a resource partition.

6.1.2 PartAmount

PartAmount provides a container for specifying amount related attributes.

Note: Multiple **PartAmount** elements are used to specify partial completion of resources. For instance, specifying **PartAmount/Part/@Side="Front"** for a **Component** would define the number of sheets that have been printed on the front side prior to printing the back side in a second press run.

Table 6.3: PartAmount Element

NAME	DATA TYPE	DESCRIPTION
<i>Amount</i> ?	float	Amount, excluding waste, in units defined in ResourceSet/@Unit or implied by ▶ Table A.3.21 Units.
<i>MaxAmount</i> ?	float	Defines the planned @Amount including the maximum overage. @MaxAmount SHALL NOT be specified as actual amounts.
<i>MinAmount</i> ?	float	Defines the planned @Amount including the maximum underage that the customer is willing to accept. @MinAmount SHALL NOT be specified as actual amounts.
<i>Waste</i> ?	float	Waste amount in units defined in ResourceSet/@Unit or implied by ▶ Table A.3.21 Units. For a resource with a @Usage of "Input", @Waste specifies the amount of the resource that MAY be consumed or has been consumed by the process. For an resource with a @Usage of "Output", @Waste specifies the amount of the resource that MAY be produced by the process.
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 uniquely specified in any parent Resource/Part element. If any of these Part elements specify the same partition key as the parent Resource/Part element, then the value of that key SHALL match one of the values from the parent Resource/Part .
PartWaste *	element	Particulars of different types and/or sources of waste MAY be specified by providing one or more PartWaste elements.

6.1.2.1 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.

6.1.3 Part

Part elements define the context in which the individual **Resource** is used. **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 all parts.

Note: The attributes of **Part** are also referred to as 'partition keys'.

Table 6.4: Part Element (Sheet 1 of 3)

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>ContactType</i> ?	NMTOKEN	@ContactType specifies the role of a contact. @ContactType SHALL be provided for ResourceSet[@Name="Contact"] . Values include those from: ▶ Contact Types.
<i>DocIndex</i> ?	IntegerRange	@DocIndex SHALL select a set of logical instance documents. The index SHALL refer to the list of documents in the context of all document sets selected by @SetIndex . Specifying @DocIndex in a RunList SHALL select individual documents without modifying the page position.

Table 6.4: Part Element (Sheet 2 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>DropID</i> ?	NMTOKEN	Identifier of an individual drop within a Delivery . A drop represents one or more items being delivered to one address at one point in time. If multiple DeliveryParams contain the same <i>@DropID</i> , they SHOULD be delivered in one delivery, regardless of whether the DeliveryParams are defined in the same XJDF or not.
<i>Location</i> ?	NMTOKEN	Name of a location. This part key allows the description of distributed ResourceSet items. Values include those from: ▶ Input Tray and Output Bin Names.
<i>LotID</i> ?	NMTOKEN	Identifier of the lot of a resource in a lot controlled environment. Examples include individual reels for web printing.
<i>Metadata</i> ?	regExp	Metadata SHALL match metadata extracted from a PDL using RunList/MetadataMap or IdentificationField/MetadataMap . See ▶ Section 8.21 MetadataMap.
<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>PartVersion</i> ?	NMTOKEN	Version identifier (e.g., the language version of a catalog).
<i>PreviewType</i> ?	enumeration	<i>@PreviewType</i> specifies the type and usage of a Preview . <i>@PreviewType</i> SHALL NOT be specified for resources other than Preview or PreviewGenerationParams . Allowed values are: Animation – Animated previews for 3D display. Identification – Preview is used as a visual help to identify one or more products, e.g. on a gang form. SeparatedThumbNail – Very low resolution separated preview. Separation – Separated preview in medium resolution. SeparationRaw – Separated preview in medium resolution with no compensation. Static3D – Static 3D model. ThumbNail – Very low resolution RGB preview. Viewable – RGB preview in medium resolution.
<i>PrintCondition</i> ?	NMTOKEN	<i>@PrintCondition</i> specifies a characterization data set that is applied to a specific setup including paper selection and screening. See ▶ Appendix A.3.16 PrintStandard Characterization Data Sets for details of characterization data sets.
<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	<i>@Run</i> identifies an individual RunList Resource .
<i>RunIndex</i> ?	IntegerRange	<i>@RunIndex</i> SHALL select a set of logical pages from a RunList resource in a manner that is independent from the internal structure of the RunList . The index SHALL refer to the list of pages in the context of all documents and sets selected by <i>@DocIndex</i> and <i>@SetIndex</i> . Specifying <i>@RunIndex</i> in a RunList SHALL select individual pages without modifying the page position.

Table 6.4: Part Element (Sheet 3 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>Separation</i> ?	NMTOKEN	Identifies a color separation. If the separation name can be represented as an NMTOKEN, the value of <i>@Separation</i> SHOULD be identical to the separation name. Otherwise the separation name SHALL be provided in <i>Resource/Color/@ActualColorName</i> of a <i>Resource</i> that contains a matching value of <i>Part/@Separation</i> . Values include: <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 "none" is spelled in lower case so that its value is identical to the pre-defined separation "none" in ▶ [PDF1.6].
<i>SetIndex</i> ?	IntegerRange	The <i>@SetIndex</i> attribute SHALL select a set of logical instance document sets. The index always refers to entries of the entire <i>RunList</i> . Specifying <i>@SetIndex</i> in a <i>RunList</i> SHALL select individual document sets without modifying the page position.
<i>SheetIndex</i> ?	IntegerRange	<i>@SheetIndex</i> selects a set of logical sheets from a <i>RunList</i> resource either implicitly or explicitly partitioned by <i>@SheetIndex</i> . <i>@SheetIndex</i> SHALL NOT be specified unless the <i>RunList</i> is describing imposed sheets or 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 <i>Part</i> element refers to one surface of the sheet. In case of web printing, "Front" is a synonym for the upper side and "Back" for the down side of the web. Allowed value is from: ▶ Side.
<i>StationName</i> ?	NMTOKEN	The name of the 1-up design in a <i>DieLayout</i> .
<i>TileID</i> ?	XYPair	XYPair of integer values that identifies a tile when a surface has been split into multiple tiles by the <i>Imposition</i> process. Values are zero-based and SHALL originate at the lower left. So "0 0" is the lower left tile and "1 0" is the tile next to it on the right.
<i>TransferCurveName</i> ?	enumeration	<i>@TransferCurveName</i> SHALL specify the destination system that the <i>TransferCurve</i> SHALL apply to. Allowed values are: <i>Film</i> – The transformation from the <i>Layout</i> system to the film. <i>Plate</i> – The transformation from the <i>Layout</i> system to the plate. <i>Press</i> – The transformation from the <i>Layout</i> system to the press. <i>Proof</i> – The transformation from the <i>Layout</i> system to the proof. <i>Substrate</i> – The transformation from the <i>Layout</i> system to the final printed substrate such as paper or plastic.
<i>WebName</i> ?	NMTOKEN	A string that uniquely identifies each web on a web press.

6.1.3.1 Partition Bootstrapping

Partition bootstrapping is the process that is employed by a consuming device to identify the *Resource* partitions that SHOULD be used when executing an *XJDF* process.

ResourceSet[@Name="NodeInfo"] defines the structure of the individual planned process steps. Thus the list of *ResourceSet*[@Name="NodeInfo"]/*Resource/Part* defines the planned partitions that SHALL be searched for each work-step. The *NodeInfo* structure MAY be a superset of the actual processes, since the planning can be less granular than the

RESOURCES

capabilities of a given device. For instance, an MIS might plan press runs on a sheet level for a non-perfecting press. In this case, each surface will be printed in one press run, and **Part/@Side** will have to be taken into account to retrieve the correct set of plates, even though **ResourceSet[@Name="NodeInfo"]/Resource/Part/@Side** is not specified. The expansion of underspecified **ResourceSet[@Name="NodeInfo"]** elements is device dependent.

Note: In general the partitioning of **NodeInfo** will correspond to the partitioning of the least granular resources. In the case of binding, input sheets will be ignored, since all sheets are bound together, whereas in the case of cutting, the output cut blocks will be ignored, since there will be a cutting process planned to cut the blocks.

6.1.3.2 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** SHALL be selected. If a single result is expected, for instance when searching for setup parameters, the iteration SHALL stop after the first match. If multiple results are expected, for instance when selecting the process color plates for a press sheet, the iteration SHALL continue for all **Resource** elements of the **ResourceSet**.

Note: There NEED NOT be a match for any given partition. This feature can be used to exclude an operation from a given partition. Thus if a process finds no matching **Resource** for a given set of partition keys, then this operation SHALL NOT be applied.

Note: Any **Resource** that contains a **Part** with no attributes or that contains no **Part** at all, SHALL always be selected.

6.1.3.3 Multiple Part Elements in One Resource

A **ResourceSet** MAY contain one or more **Resource** elements that MAY respectively contain zero 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 Cyan, Magenta and Yellow would each contain 2 **Part** elements with both English and French whereas the two individual **Resource** elements for the Black plates would contain the respective individual **@PartVersion**.

Example 6.1: 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 MediaRef="EM42"/>
  </Resource>
  <Resource>
    <Part Separation="Magenta" PartVersion="English"/>
    <Part Separation="Magenta" PartVersion="French"/>
    <ExposedMedia MediaRef="EM42"/>
  </Resource>
  <Resource>
    <Part Separation="Yellow" PartVersion="English"/>
    <Part Separation="Yellow" PartVersion="French"/>
    <ExposedMedia MediaRef="EM42"/>
  </Resource>
  <!-- Specific Black Plate for English -->
  <Resource>
    <Part Separation="Black" PartVersion="English"/>
    <ExposedMedia MediaRef="EM42"/>
  </Resource>
  <!-- Specific Black Plate for French -->
  <Resource>
    <Part Separation="Black" PartVersion="French"/>
    <ExposedMedia MediaRef="EM42"/>
  </Resource>
</ResourceSet>
```

6.1.4 PartWaste

PartWaste associates waste with individual device modules or waste types.

Note: The sum of specific waste can be higher than the total waste due to double counting.

Table 6.5: PartWaste Element

NAME	DATA TYPE	DESCRIPTION
<i>ModuleIds</i> ?	NMTOKENS	Specifies the module or modules where the waste was produced.
<i>Waste</i>	float	Specific waste amount that SHALL be in the same units as those of the parent <i>PartAmount</i> .
<i>WasteDetails</i> ?	NMTOKEN	@ <i>WasteDetails</i> specifies additional details about how the waste was produced. See ▶ Table 6.6 WasteDetails Attribute Values for suggested values. At least one of @ <i>ModuleIds</i> or @ <i>WasteDetails</i> SHALL be specified.

Table 6.6: WasteDetails Attribute Values

VALUE	DESCRIPTION
<i>AuxiliarySheet</i>	This value identifies <i>InsertSheet</i> media that was consumed as specified by <i>StackingParams/Disjointing/InsertSheet</i> .
<i>BadFeedWaste</i>	Waste caused by a bad feed.
<i>BindingQualityTest</i>	<i>Components</i> that were destroyed in a <i>QualityControl</i> process that tests binding quality. Additional information about failed and passed tests SHOULD be provided in <i>QualityControlResult</i> .
<i>CaliperWaste</i>	Waste by caliper on gathering / collecting.
<i>DoubleFeedWaste</i>	Waste by double feeds on feeders.
<i>IncorrectComponentWaste</i>	Waste by the attempted use of an incorrect component (e.g., on a feeder).
<i>ObliqueSheetWaste</i>	Waste by oblique sheets on gathering / collecting chains.
<i>Overrun</i>	Excess <i>Component</i> resource(s) that were produced by running the device after the specified amount had been produced.
<i>PaperJamWaste</i>	Waste by paper or other media jam.
<i>Rejected</i>	Rejected in an approval process.
<i>Reusable</i>	Waste to be used for setup in the next process.
<i>Waste</i>	General waste.
<i>WhitePaperWaste</i>	White paper waste.

6.2 ApprovalDetails

The signed *ApprovalDetails* resource indicates whether a resource has been approved or rejected.

Resource Properties

Intent Pairing: *ContentCheckIntent*

Input of Processes: *Any Process*

Output of Processes: *Approval*

Table 6.7: ApprovalDetails Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ApprovalState</i>	enumeration	Decision made by the approver. Allowed values are: <i>Approved</i> – Approver approved the resource. <i>ApprovedWithComment</i> – Approver approved the resource but still had some comments. <i>Rejected</i> – Approver rejected the resource.

Table 6.7: ApprovalDetails Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ApprovalStateDetails</i> ?	string	Additional details on the decision made by the approver are specified in this <i>@ApprovalStateDetails</i> . This value provides additional machine readable details of <i>@ApprovalState</i> . Hand written comments and notes MAY be specified in <i>ApprovalDetails/Comment</i> .
<i>ApprovalPerson</i> ?	element	Details of the person (e.g., a customer, printer or manager) who processed the approval.
<i>Comment</i> ?	element	This <i>Comment</i> provides a container for human readable notes that are provided by the approver.
<i>FileSpec</i> ?	element	The file that contains the approval signature.

6.3 ApprovalParams

ApprovalParams provides the details of an *Approval* process.

Resource Properties

Intent Pairing: *ContentCheckIntent*

Input of Processes: *Approval*

Table 6.8: ApprovalParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>ApprovalPerson</i> +	element	List of people (e.g., a customer, printer or manager) who can sign the approval.

6.4 Assembly

Assembly describes how multiple *BinderySignatures* are bound together to produce a bound product.

Resource Properties

Intent Pairing: *LayoutIntent*

Input of Processes: *Collecting, Gathering, SheetOptimizing, Stripping, WebInlineFinishing*

Table 6.9: Assembly Resource

NAME	DATA TYPE	DESCRIPTION
<i>BinderySignatureIDs</i> ?	NMTOKENS	<i>@BinderySignatureIDs</i> specifies an ordered list of <i>BinderySignature</i> that SHALL be assembled by the method specified in <i>@Order</i> . <i>@BinderySignatureIDs</i> SHALL NOT be present, if <i>@Order="List"</i> .
<i>Order</i>	enumeration	Ordering of the individual <i>BinderySignature</i> elements. Order specifies the topology of the final <i>Assembly</i> . Allowed values are: <i>Collecting</i> – The sections are placed within one another. The first <i>BinderySignature</i> specified in <i>@BinderySignatureIDs</i> is on the outside. An example is a saddle-stitched brochure. See ▶ Section 5.6.7 Collecting. <i>Gathering</i> – The sections are placed on top of one another. The first <i>BinderySignature</i> specified in <i>@BinderySignatureIDs</i> is on the top. An example is a perfect bound magazine. See ▶ Section 5.6.16 Gathering. <i>List</i> – The <i>Assembly</i> SHALL be fully described with <i>AssemblySection</i> elements. <i>None</i> – The sections are not bound. Typically used for flatwork jobs.
<i>AssemblySection</i> *	element	Each <i>AssemblySection</i> element represents one section that SHALL be gathered. The first <i>AssemblySection</i> SHALL be placed on top, i.e. it is the front of the <i>Assembly</i> . <i>AssemblySection</i> elements SHALL NOT be specified unless <i>@Order="List"</i> and SHALL be specified if <i>@Order="List"</i> .

6.4.1 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 6.10: AssemblySection Element

NAME	DATA TYPE	DESCRIPTION
<i>BinderySignatureID</i>	NMTOKEN	@ <i>BinderySignatureID</i> identifies the BinderySignature that this AssemblySection represents.
AssemblySection *	element	Additional child AssemblySection elements that SHALL be gathered. The resulting set of AssemblySection elements SHALL be collected inside this AssemblySection .

Example 6.2: Perfect Bound (Gathering)

Cover wrapped around a perfect bound (gathering) body.

```
<ResourceSet Name="Assembly">
  <Resource>
    <Assembly Order="List">
      <AssemblySection BinderySignatureID="Ass_Cover">
        <AssemblySection BinderySignatureID="Ass_Body1"/>
        <AssemblySection BinderySignatureID="Ass_Body2"/>
        <AssemblySection BinderySignatureID="Ass_Insert"/>
        <AssemblySection BinderySignatureID="Ass_Body3"/>
        <AssemblySection BinderySignatureID="Ass_Body4"/>
      </AssemblySection>
    </Assembly>
  </Resource>
</ResourceSet>
```

Example 6.3: Saddle-Stitched Brochure (Collecting)

```
<ResourceSet Name="Assembly">
  <Resource>
    <Assembly Order="List">
      <AssemblySection BinderySignatureID="Ass_Cover">
        <AssemblySection BinderySignatureID="Ass_Body1">
          <AssemblySection BinderySignatureID="Ass_Body2">
            <AssemblySection BinderySignatureID="Ass_Body3">
              <AssemblySection BinderySignatureID="Ass_Body4"/>
            </AssemblySection>
          </AssemblySection>
        </AssemblySection>
      </AssemblySection>
    </AssemblySection>
  </AssemblySection>
</Resource>
</ResourceSet>
```

6.5 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

Input of Processes: **Bending**

Table 6.11: BendingParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Bend</i> ?	boolean	If "true", indicates that the device SHALL bend.

Table 6.11: *BendingParams Resource (Sheet 2 of 2)*

NAME	DATA TYPE	DESCRIPTION
<i>Punch</i> ?	boolean	If "true", indicates that the device SHALL create registration punch holes.
<i>PunchType</i> ?	NMTOKEN	Name of the registration punch scheme (e.g., Bacher).

6.6 BinderySignature

A *BinderySignature* represents both sides of a folding signature, a die cut surface or a flat product such as a postcard, each with one or more pages. *Resource/Part/@BinderySignatureID* SHALL be provided for a *BinderySignature*. Multiple *Resource/Part* elements MAY be provided, in which case the *BinderySignature* element represents multiple physical bindery signatures.

Resource Properties

Intent Pairing: *LayoutIntent*

Input of Processes: *SheetOptimizing, Stripping*

Table 6.12: *BinderySignature Resource (Sheet 1 of 2)*

NAME	DATA TYPE	DESCRIPTION
<i>BinderySignatureSize</i> ?	XYPair	Size of the <i>BinderySignature</i> . <i>@BinderySignatureSize</i> SHALL be identical to the sum of sizes including trim of all <i>SignatureCells</i> .
<i>BinderySignatureType</i> ?	enumeration	<i>@BinderySignatureType</i> specifies the type of <i>BinderySignature</i> and the pagination rules. Allowed values are: <i>Die</i> – A layout defined by an existing die. <i>Fold</i> – The <i>BinderySignature</i> represents a folding signature. The pagination SHALL be calculated from the <i>Assembly, Layout</i> and <i>BinderySignature</i> . See ▶ Appendix F Pagination Catalog. <i>Grid</i> – The <i>BinderySignature</i> represents a grid based layout. The pagination SHALL be explicitly specified in <i>SignatureCell/@FrontPages</i> and <i>SignatureCell/@BackPages</i> . Grid SHALL be used for flatwork.
<i>BindingOrientation</i> ?	enumeration	After folding a <i>BinderySignature</i> , the default coordinate system SHALL be the coordinate system with the binding edge on the left side and the jog edge at the top. <i>@BindingOrientation</i> defines the transformation that SHALL be applied to the <i>BinderySignature</i> prior to calculating pagination. For example, a value of "Rotate180" would rotate every page by 180° but retain the pagination whereas a value of "Flip180" would reverse the pagination while retaining the page orientation. Note: In some cases, the first page of a <i>BinderySignature</i> is placed on the back side of the sheet. For details, see ▶ Table 2.1 Matrices and Orientation values for describing the orientation of a Component. Value is from: ▶ Orientation.
<i>Bottling</i> ?	enumeration	<i>@Bottling</i> 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 <i>DieLayout</i> . <i>DieLayout</i> SHALL NOT be present unless <i>@BinderySignatureType="Die"</i> .
<i>FoldCatalog</i> ?	NMTOKEN	<i>@FoldCatalog</i> describes folding of the <i>BinderySignature</i> . At least one of <i>SignatureCell, @FoldCatalog</i> or <i>@DieLayoutRef</i> SHALL be specified. Values include those from: ▶ Fold Catalog.

Table 6.12: BinderySignature Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<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 SHALL be positive integers).
<i>Overfold</i> ?	float	Size of the overfold.
<i>OverfoldSide</i> ?	enumeration	Position of the overfold in the finished signature. Allowed values are: Back - The overfold is on the back side. All pages have an overfold. BackHalf - The overfold is on the back side. Only pages on one side of the unfolded signature have an overfold. Front - The overfold is on the front side. All pages have an overfold. FrontHalf - The overfold is on the front side. Only pages on one side of the unfolded signature have an overfold.
<i>StaggerColumns</i> ?	FloatList	A list of values 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 <i>SignatureCell</i> height ((y value of <i>@TrimSize</i>) + <i>@TrimHead</i> + <i>@TrimFoot</i>) by which to shift the corresponding column. Note: Each value MAY 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 down by 33.3% of the <i>SignatureCell</i> height • 3*n+2 column up by 66.6% of the <i>SignatureCell</i> 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.
<i>StaggerContinuous</i> ?	boolean	Indicates if the <i>BinderySignature</i> SHALL 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 <i>m</i> columns and <i>n</i> rows with <i>@StaggerContinuous</i> ="true". If <i>@StaggerColumns</i> is specified, the <i>BinderySignature</i> SHALL be considered continuous with a height <i>H</i> equal to <i>n</i> multiplied by the <i>SignatureCell</i> height. If <i>@StaggerColumns</i> has a value of <i>y</i> for a certain column, that column is shifted up (assuming <i>y</i> > 0) by an amount equal to <i>y</i> multiplied by the <i>SignatureCell</i> height (in the same way as described for <i>@StaggerColumns</i>). All content (even partial cells) that falls above <i>H</i> (the top of <i>BinderySignature</i>) 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 <i>y</i> 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> ?	FloatList	A list of values 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 <i>SignatureCell</i> width by which to shift the corresponding row. Note: Each value MAY 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 <i>SignatureCell</i> width • 3*n+2 row left by 66.6% of the <i>SignatureCell</i> 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>SignatureCell</i> *	element	Describes the <i>SignatureCell</i> elements used in this <i>BinderySignature</i> . <i>SignatureCell</i> elements are ordered in X-Y direction starting at the lower left corner of the <i>BinderySignature</i> . When both <i>SignatureCell</i> and <i>@FoldCatalog</i> are specified, <i>@FoldCatalog</i> defines the topology of the folding scheme, and the specifics of each individual signature cell are described by the <i>SignatureCell</i> elements. The <i>SignatureCell</i> elements SHALL have precedence.

6.6.1 SignatureCell

SignatureCell elements describe the geometry of one or more individual page cells in a **BinderySignature**.

The width (horizontal size) of a **SignatureCell** SHALL be calculated as **@TrimFace** + **@TrimSpine** + the X value of **@TrimSize**. If **BinderySignature/@Overfold** is specified in the parent **BinderySignature** and the **SignatureCell** applies to a page that contains an overfold according to the value of the parent **BinderySignature/@Overfold**, then the value of **BinderySignature/@Overfold** SHALL be added to the width of the **SignatureCell**. The height (vertical size) of a **SignatureCell** SHALL be calculated as **@TrimHead** + **@TrimFoot** + the Y value of **@TrimSize**. See ▶ Figure 6-1: Definition of margins in SignatureCell for details.

If no **SignatureCell** is specified, the exact margins should be calculated by the stripping process.

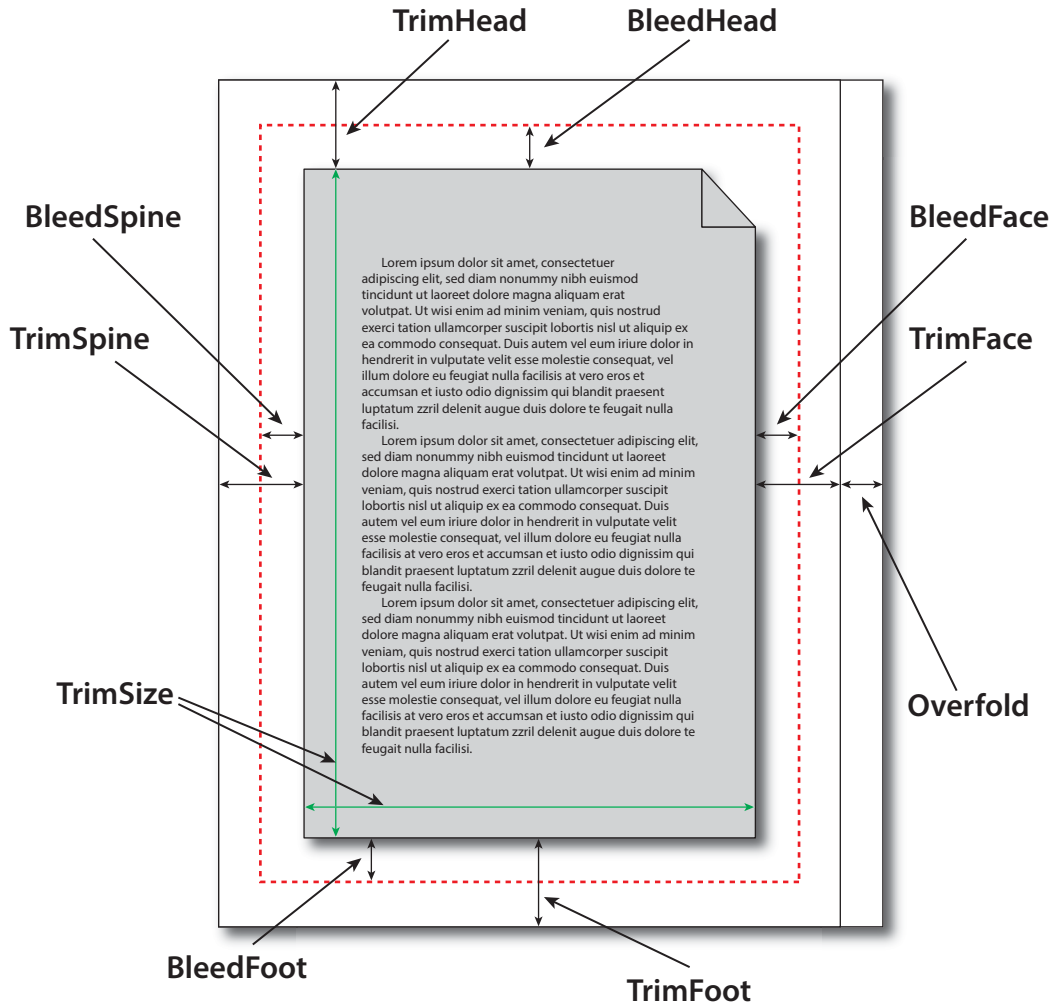
Table 6.13: SignatureCell Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>BackPages</i> ?	IntegerList	Page numbers of the back pages of a SignatureCell . The number of entries in @FrontPages and @BackPages SHALL be identical. The entries with an identical index in @FrontPages and @BackPages are back-to-back in the layout. If not specified, the layout is one-sided.
<i>BackSpread</i> ?	IntegerList	List of indices of SignatureCell elements that are combined into a spread on the back side.
<i>BleedFace</i> ?	float	Amount of bleed that SHALL be added to the box defined in @TrimBox at the face side.
<i>BleedFoot</i> ?	float	Amount of bleed that SHALL be added to the box defined in @TrimBox at the foot side.
<i>BleedHead</i> ?	float	Amount of bleed that SHALL be added to the box defined in @TrimBox at the head side.
<i>BleedSpine</i> ?	float	Amount of bleed that SHALL be added to the box defined in @TrimBox at the spine side.
<i>FaceCells</i> ?	IntegerList	List of indices of SignatureCell elements that form a foldout together with this SignatureCell . The SignatureCell that contains @FaceCells 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>FrontPages</i> ?	IntegerList	Page numbers of the front pages of a SignatureCell . Multiple page cells with the same properties except for the pages to which they are assigned MAY be summarized as one SignatureCell with multiple entries in @FrontPages .
<i>FrontSpread</i> ?	IntegerList	List of indices of SignatureCell elements that are combined into a spread on the front side.
<i>Mask</i> ?	enumeration	The definition of the clipping mask for the placed graphics. Allowed values are: BleedBox – The mask is derived from the bleed box as defined by the SignatureCell . DieCut – The mask SHALL be derived from the cut line as defined in DieLayout/@CutLines . None – No mask. PDL – The mask is derived from the PDL of the graphics. @MaskSeparation MAY determine which separation SHALL be used as the clipping mask for the graphics. If @MaskSeparation is not present, the mask SHALL be determined from the underlying PDL, e.g. ▶ [ISO19593-1:2016]. SourceBleedBox – The mask is derived from the bleed box of the graphical element placed in the SignatureCell . SourceTrimBox – The mask is derived from the trim box of the graphical element placed in the SignatureCell . TrimBox – The mask is derived from the trim box as defined by the SignatureCell .
<i>MaskBleed</i> ?	float	The distance over which to expand the mask in points.

Table 6.13: SignatureCell Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>MaskSeparation</i> ?	NMTOKEN	<i>Color</i> ././ <i>Resource/Part</i> /@ <i>Separation</i> of the <i>Color</i> that specifies @ <i>Mask</i> . @ <i>MaskSeparation</i> SHALL NOT be specified unless @ <i>Mask</i> ="PDL". <i>Color</i> /@ <i>ColorType</i> of this separation SHALL be "DieLine".
<i>Orientation</i> ?	enumeration	Indicates the orientation of the <i>SignatureCell</i> on the <i>BinderySignature</i> . 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> ?	enumeration	Sides SHALL specify which side of the media SHALL be printed. Allowed value is from: ▶ <i>Sides</i> .
<i>StationName</i> ?	NMTOKEN	The name of the 1-up station in the die layout. If <i>BinderySignature</i> /@ <i>BinderySignatureType</i> ="Die", this element SHOULD be specified. If <i>BinderySignature</i> /@ <i>BinderySignatureType</i> ="Die" and the <i>DieLayout</i> referenced by <i>BinderySignature</i> /@ <i>DieLayoutRef</i> contains more than 1 <i>Station</i> , this attribute SHALL be specified.
<i>TrimFace</i> ?	float	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>TrimSpine</i> .
<i>TrimFoot</i> ?	float	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> ?	float	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 F Pagination Catalog.
<i>TrimSize</i> ?	XYPair	@ <i>TrimSize</i> defines the dimensions of the trim box. If not specified, @ <i>TrimSize</i> SHALL be extracted from the PDL that is referenced from the input <i>RunList</i> (<i>Document</i>) of the Stripping or Imposition process.
<i>TrimSpine</i> ?	float	Amount of paper that 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>TrimSpine</i> . Note: See ▶ Appendix F Pagination Catalog.

Figure 6-1: Definition of margins in SignatureCell



Note: Overfold in this figure refers to the value of the parent [BinderySignature/@Overfold](#) and is only present if the page requires an overfold based on the value of the parent [BinderySignature/@OverfoldSide](#).

6.7 BlockPreparationParams

[BlockPreparationParams](#) describes the settings of a [BlockPreparation](#) process.

Resource Properties

Intent Pairing: [BindingIntent](#)

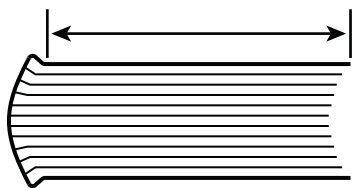
Input of Processes: [BlockPreparation](#)

Table 6.14: [BlockPreparationParams](#) Resource

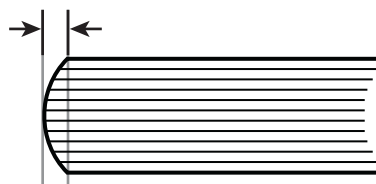
NAME	DATA TYPE	DESCRIPTION
Backing ?	float	Backing distance in points. See @Backing : ▶ Figure 6-2: Backing and Rounding measurements for TightBacking.
Rounding ?	float	Rounding distance in points. See @Rounding : ▶ Figure 6-2: Backing and Rounding measurements for TightBacking.
TightBacking ?	enumeration	Definition of the geometry of the back of the book block. Allowed value is from: ▶ TightBacking.
RegisterRibbon *	element	Description of the register ribbons that are included within the book block.

Figure 6-2: Backing and Rounding measurements for TightBacking

@Backing



@Rounding



6.8 BoxFoldingParams

BoxFoldingParams defines the parameters for folding and gluing blanks to folded flat boxes in a box folder-gluer device.

Resource Properties

Input of Processes: **BoxFolding**

Table 6.15: BoxFoldingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>BlankDimensionsX</i> ?	FloatList	<i>@BlankDimensionsX</i> contains a list of the X positions of folds for an unfolded box beginning from the origin of the coordinate system (left side) increasing from minimum to maximum. These positions are described in ▶ Section 6.8.1 BoxFoldingType attribute values. <i>@BlankDimensionsX</i> SHOULD be specified if BoxFoldAction elements are specified.
<i>BlankDimensionsY</i> ?	FloatList	<i>@BlankDimensionsY</i> contains a list of the Y positions of folds for an unfolded box beginning from the origin of the coordinate system (bottom side) increasing from minimum to maximum. These positions are described in ▶ Section 6.8.1 BoxFoldingType attribute values. <i>@BlankDimensionsY</i> SHOULD be specified if BoxFoldAction elements are specified.
<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 MAY be described with a sequence of BoxFoldAction and Glue elements. If <i>@BoxFoldingType</i> ="Type00", the sequence of BoxFoldAction and Glue elements SHALL be provided. If <i>@BoxFoldingType</i> !="Type00", the sequence of BoxFoldAction and Glue elements MAY be provided, and if provided the sequence of BoxFoldAction elements shall match the appropriate sequence provided in ▶ Table 6.16 BoxFoldingType Attribute Values. Allowed values are from: ▶ Table 6.16 BoxFoldingType Attribute Values.
<i>BoxFoldAction</i> *	element	Individual work step in a box folder-gluer. The sequence of BoxFoldAction and Glue elements SHALL define the sequence of work steps and MAY occur in any order. The first element SHALL be applied first. ^a
<i>Glue</i> *	element	Specification of a glue line. The Glue is applied to the blank in the coordinate system of the folder gluer at the state after all prior BoxFoldActions have been applied. The sequence of BoxFoldAction and Glue elements defines the sequence of work steps and MAY occur in any order. The first element SHALL be applied first. ^a

- a. The order of **BoxFoldAction** and **Glue** elements precisely determines the sequence of operations. Hence the normal XJDF rules for ordering XML elements does not apply.

6.8.1 BoxFoldingType attribute values

The following table shows the allowed values for *@BoxFoldingType*. Each type corresponds to a particular style of folding shown in the figure column. The description column contains a list of 'dimension/action' for each fold. Dimension references a specific value from **BoxFoldingParams** where X0 is the first value from *@BlankDimensionsX*, and Y0 is the first value from *@BlankDimensionsY*. X1/Y1 refer to the appropriate second value etc. Action references a specific value from **BoxFoldingParams/BoxFoldAction/@Action**, see ▶ Table 6.18 Action Attribute Values.

Notes:

- 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

Table 6.16: BoxFoldingType Attribute Values

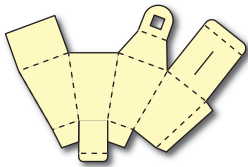
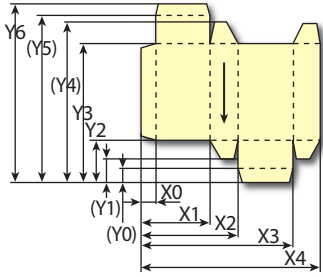
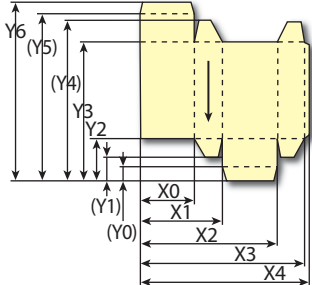
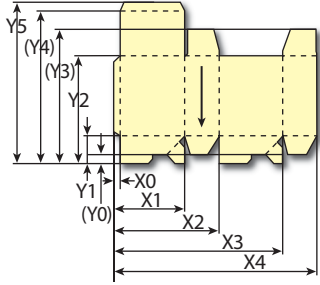
VALUE	FIGURE	DESCRIPTION
Type00		Special type for boxes that are not pre-defined.
Type01		X0 LongPreFoldLeftToRight X2 LongPreFoldRightToLeft X1 LongFoldLeftToRight X3 LongFoldRightToLeft
Type02		X3 LongPreFoldLeftToRight X1 LongPreFoldRightToLeft X0 LongFoldLeftToRight X2 LongFoldRightToLeft
Type03		X0 LongPreFoldLeftToRight X2 LongPreFoldRightToLeft X2/Y1 FrontFoldComplete X4/Y1 FrontFoldComplete X1/Y1 FrontFoldCompleteDiagonal X3/Y1 FrontFoldCompleteDiagonal X1 LongFoldLeftToRight X3 LongFoldRightToLeft

Table 6.16: BoxFoldingType Attribute Values

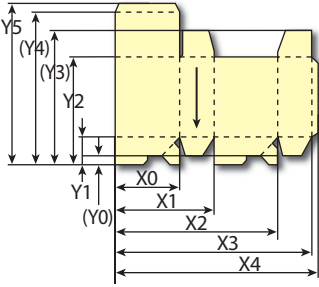
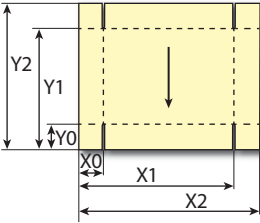
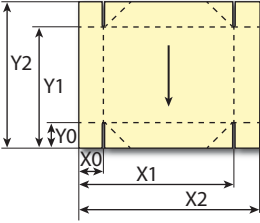
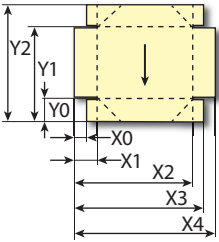
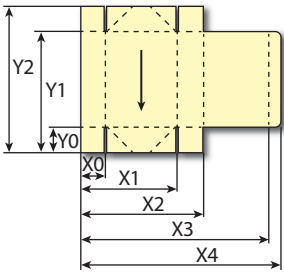
VALUE	FIGURE	DESCRIPTION
Type04		<p>X3 LongPreFoldLeftToRight X1 LongPreFoldRightToLeft X1/Y1 FrontFoldComplete X3/Y1 FrontFoldComplete X0/Y1 FrontFoldCompleteDiagonal X2/Y1 FrontFoldCompleteDiagonal X0 LongFoldLeftToRight X2 LongFoldRightToLeft</p>
Type10		<p>X0 LongPreFoldLeftToRight X1 LongPreFoldRightToLeft</p>
Type11		<p>X0/Y0 FrontFoldComplete X2/Y0 FrontFoldComplete X0/Y2 BackFoldComplete X2/Y2 BackFoldComplete X1/Y0 FrontFoldCompleteDiagonal X1/Y2 BackFoldCompleteDiagonal X0 LongFoldLeftToRight X2 LongFoldRightToLeft</p>
Type12		<p>X1/Y0 FrontFoldCompleteDiagonal X1/Y2 BackFoldCompleteDiagonal X0 LongFoldLeftToRight X2 LongFoldRightToLeft</p>
Type13		<p>X0/Y0 FrontFoldComplete X2/Y0 FrontFoldComplete X0/Y2 BackFoldComplete X2/Y2 BackFoldComplete X1/Y0 FrontFoldCompleteDiagonal X1/Y2 BackFoldCompleteDiagonal X0 LongFoldLeftToRight X2 LongFoldRightToLeft</p>

Table 6.16: BoxFoldingType Attribute Values

VALUE	FIGURE	DESCRIPTION
Type15		<p>X0/Y0 FrontFoldComplete X2/Y0 FrontFoldComplete X4/Y0 FrontFoldComplete X0/Y2 BackFoldComplete X2/Y2 BackFoldComplete X4/Y2 BackFoldComplete X1/Y0 FrontFoldCompleteDiagonal X1/Y0 FrontFoldCompleteDiagonal X1/Y2 BackFoldCompleteDiagonal X3/Y2 BackFoldCompleteDiagonal X0 LongFoldLeftToRight X3 LongFoldRightToLeft X2 LongFoldRightToLeft</p>
Type20		<p>X0 LongFoldLeftToRight X3 LongFoldRightToLeft</p>

6.8.2 BoxFoldAction

BoxFoldAction describes an action in the folder-gluer that is perpendicular or diagonal to the movement path of the blank.

Table 6.17: BoxFoldAction Element

NAME	DATA TYPE	DESCRIPTION
Action	enumeration	<p>@Action describes an individual action in the folder gluer. See ▶ Figure 6-3: Folding examples for some values of BoxFoldAction/@Action. Allowed values are from: ▶ Table 6.18 Action Attribute Values.</p>
FoldIndex	XYPair	<p>Pair of indices that identify the upper right corner of the flap or fold that is affected by this BoxFoldAction. The first value of the XYPair SHALL refer to an indexed fold that is labeled X0...Xn in the specific fold template that is selected by BoxFoldingParams/@BoxFoldingType as shown in ▶ Table 6.16 BoxFoldingType Attribute Values. The second value of the XYPair shall refer to an indexed fold that is labeled Y0...Yn in the specific fold template that is selected by BoxFoldingParams/@BoxFoldingType as shown in ▶ Table 6.16 BoxFoldingType Attribute Values. If either X or Y spans multiple flaps, it SHALL be set to -1.</p>

Table 6.18: Action Attribute Values (Sheet 1 of 2)

VALUE	DESCRIPTION
LongFoldLeftToRight	
LongFoldRightToLeft	
LongPreFoldLeftToRight	
LongPreFoldRightToLeft	

Table 6.18: Action Attribute Values (Sheet 2 of 2)

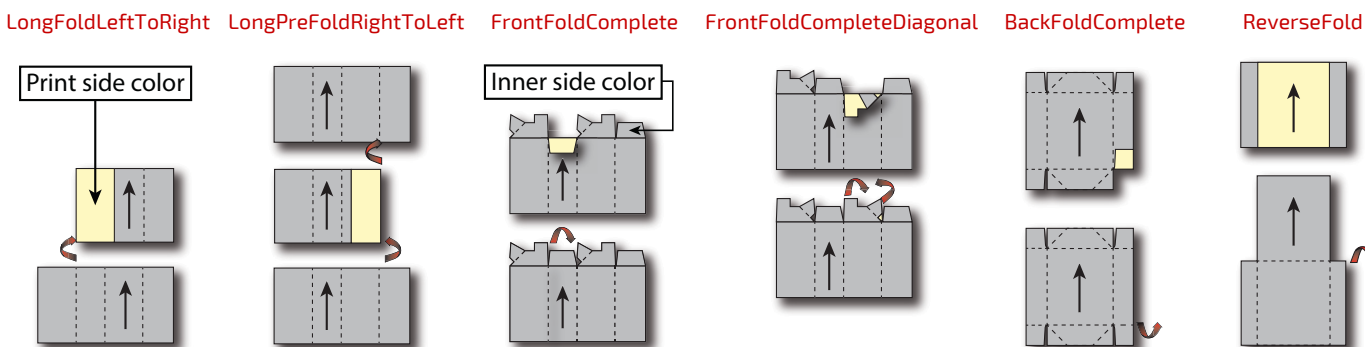
VALUE	DESCRIPTION
FrontFoldComplete	
FrontFoldDiagonal	
FrontFoldCompleteDiagonal	
BackFoldComplete	
BackFoldDiagonal	
BackFoldCompleteDiagonal	
ReverseFold	A "ReverseFold" is topologically equivalent to "FrontFoldDiagonal" but uses different equipment with other restrictions on Media weight and size and is therefore specified individually.
Milling	
Rotate90	90° counter-clockwise rotation
Rotate180	180° rotation
Rotate270	90° clockwise rotation

Example 6.4: BoxFoldingParams/BoxFoldAction

For instance, processing a Type01 blank has the following actions:

```
<ResourceSet Name="BoxFoldingParams" Usage="Input">
  <Resource>
    <BoxFoldingParams BlankDimensionsX="28 142 198 312 369"
      BlankDimensionsY="57 85 142 425 482 510 567" BoxFoldingType="Type01">
      <BoxFoldAction Action="LongPreFoldLeftToRight" FoldIndex="0 -1"/>
      <BoxFoldAction Action="LongPreFoldRightToLeft" FoldIndex="2 -1"/>
      <BoxFoldAction Action="LongFoldLeftToRight" FoldIndex="1 -1"/>
      <BoxFoldAction Action="LongFoldRightToLeft" FoldIndex="3 -1"/>
    </BoxFoldingParams>
  </Resource>
</ResourceSet>
```

Figure 6-3: Folding examples for some values of BoxFoldAction/@Action



6.9 BoxPackingParams

BoxPackingParams 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.

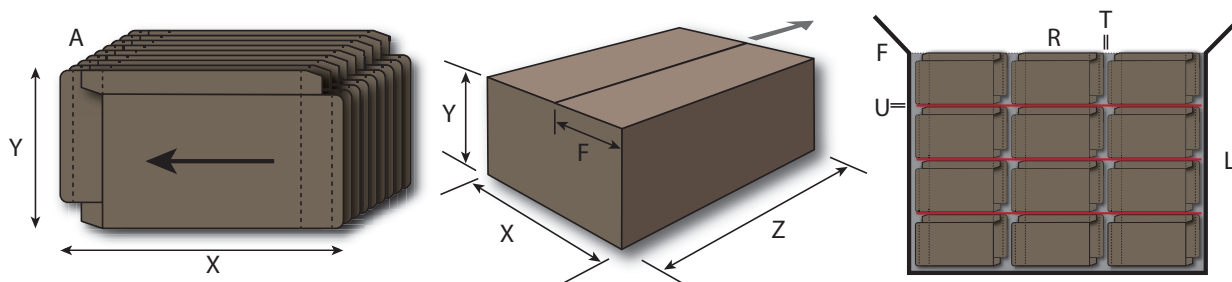
Resource Properties

Input of Processes: **BoxPacking**

Table 6.19: BoxPackingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>BoxType</i>	enumeration	<i>@BoxType</i> specifies the general category of the package to be packed. Allowed values are: Box – Boxes are convenience packaging and are not envisioned to be protection for shipping. Carton – Cartons envisioned to be protection for shipping. Envelope – Envelopes are packages that are envisioned for shipping. Tube – Tubes are cylinder shaped cartons that are envisioned for shipping.
<i>BoxTypeDetails</i> ?	string	Additional details of <i>@BoxType</i> . <i>@BoxType</i> MAY be a site specific identifier. Values include: Branded Carton Easter Bunny Box Neutral Carton
<i>Columns</i> ?	integer	Columns per shipping box. Columns are in the 3rd Dimension in ▶ Figure 6-4: Box packing, and are thus not illustrated.
<i>ComponentsPerRow</i> ?	integer	Components or Products per row in the shipping box, as illustrated by A in ▶ Figure 6-4: Box packing. If the Components represent Bundles , the number of Bundles SHALL be specified.
<i>Copies</i> ?	integer	Number of copies in the box. <i>@Copies</i> SHALL NOT be specified if <i>@MaxWeight</i> is present.
<i>FaceDown</i> ?	enumeration	Defines the surface that is facing the bottom of the box, defining the horizontal plane. Allowed values are from: ▶ Face.
<i>Layers</i> ?	integer	Layers per shipping box, as illustrated by L in ▶ Figure 6-4: Box packing.
<i>MaxWeight</i> ?	float	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 6-4: Box packing.
<i>Ties</i> ?	IntegerList	Number of tie sheets at each row, as illustrated by T in ▶ Figure 6-4: Box packing. 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 SHALL restart at the 0 position. If fewer rows than values are specified, all tie sheets that are not adjacent to a row SHALL be ignored.
<i>UnderLays</i> ?	IntegerList	Number of underlay sheets at each layer, as illustrated by U in ▶ Figure 6-4: Box packing. 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 SHALL restart at the 0 position. If fewer layers than values are specified, all underlay sheets that are not adjacent to a layer SHALL be ignored.

Figure 6-4: Box packing



6.10 Bundle

Bundles are used to describe various kinds of sets of packing units such as boxes, cartons and pallets.

The coordinate system of a **Bundle** is defined by the orientation of the first, i.e. bottom item in the **Bundle**. See ▶ Figure 6-5: Bundle coordinate system.

Resource Properties

Input of Processes: **BoxPacking, Bundling, Labeling, Palletizing, Shrinking, Stacking, Strapping, Wrapping**

Table 6.20: Bundle Resource

NAME	DATA TYPE	DESCRIPTION
BundleItem +	element	References to the individual BundleItems that form this Bundle .

6.10.1 BundleItem

A **Bundle** is described as a set of **BundleItem** elements. **BundleItem** elements describe packages that MAY reference other **BundleItem** elements which themselves MAY reference further **BundleItem** elements.

Table 6.21: BundleItem Element (Sheet 1 of 2)

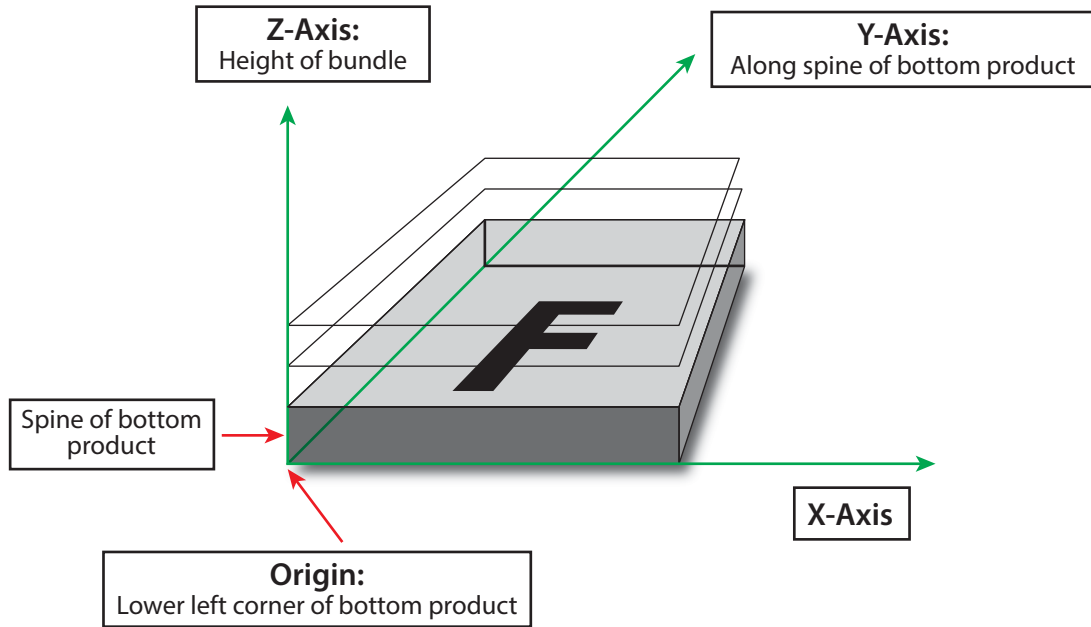
NAME	DATA TYPE	DESCRIPTION
Amount	integer	Number of identical bundle items of the same type and structure that this BundleItem represents.
BundleType ?	enumeration	@ BundleType defines the type of bundle that the BundleItem represents. Allowed values are from: ▶ BundleType . Note: @ BundleType ="Product" SHALL NOT be specified if a child BundleItem element is present.
ItemRef ?	IDREF	Reference to an individual Component or Product that is represented by this BundleItem . @ ItemRef SHALL NOT be specified if a child BundleItem element is present.
TotalAmount ?	integer	Total amount of individual products that this BundleItem contains. If @ Amount !=1, then @ TotalAmount refers to the total amount that is contained in all bundle items that the BundleItem represents. If the BundleItem or its descendent BundleItem elements contain one or more @ ItemRef attributes that reference a ProductList/Product , then @ TotalAmount refers to the number of final products.
TotalDimensions ?	shape	Total dimensions in points of all individual items including packaging that belong to this BundleItem .
TotalVolume ?	float	Total volume in liters of all individual items including packaging that belong to this BundleItem .
TotalWeight ?	float	Total weight in grams of all individual items including packaging that belong to this BundleItem .

RESOURCES

Table 6.21: BundleItem Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>BundleItem</i> *	element	Individual <i>BundleItem</i> elements that this parent <i>BundleItem</i> contains. All <i>BundleItem</i> elements in one level of the <i>BundleItem</i> tree SHOULD have the same value of <i>BundleItem</i> /@BundleType.

Figure 6-5: Bundle coordinate system



Example 6.5:

The following example **XJDF** describes 10 pallets with 420 books in 10 boxes of 42 books each.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="Bundle" Types="Palletizing">
  <ProductList>
    <Product Amount="4200" ID="BookProductID" IsRoot="true"/>
  </ProductList>
  <ResourceSet Name="Component" Usage="Input">
    <Resource GrossWeight="650">
      <AmountPool>
        <PartAmount Amount="4200"/>
      </AmountPool>
      <Part ProductPart="BookProductID"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Component" Usage="Output">
    <Resource GrossWeight="50300">
      <AmountPool>
        <PartAmount Amount="10"/>
      </AmountPool>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Bundle" Usage="Input">
    <Resource>
      <Bundle>
        <BundleItem Amount="10" BundleType="Pallet" TotalAmount="4200">
          <BundleItem Amount="10" BundleType="Carton"
            ItemRef="BookProductID" TotalAmount="420"/>
        </BundleItem>
      </Bundle>
    </Resource>
  </ResourceSet>
</XJDF>
```

6.11 BundlingParams

BundlingParams describes the details of a **Bundling** process.

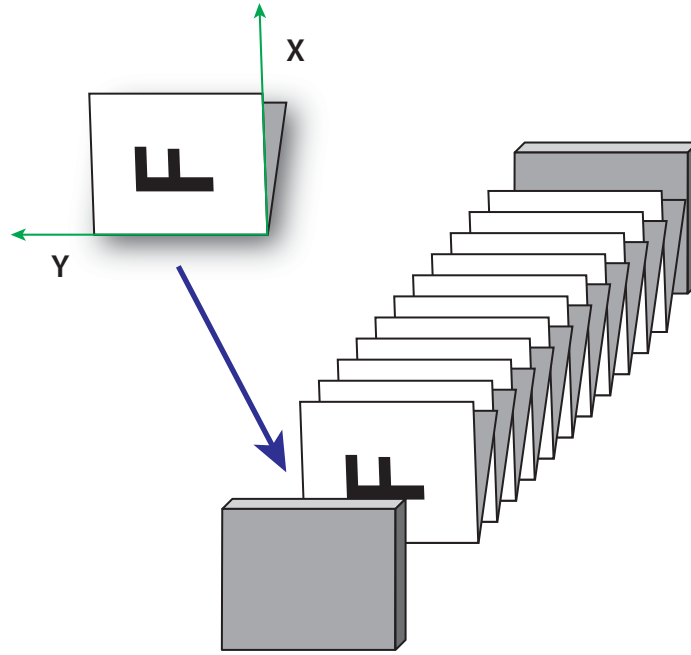
Resource Properties

Input of Processes: **Bundling**

Table 6.22: **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> ?	float	Length of a bundle. <i>@Length</i> SHALL NOT be specified if <i>@Copies</i> is present.

Figure 6-6: BundlingParams coordinate system



6.12 CaseMakingParams

CaseMakingParams describes the settings of a **CaseMaking** process for hardcover binding.

Resource Properties

Intent Pairing: **BindingIntent**

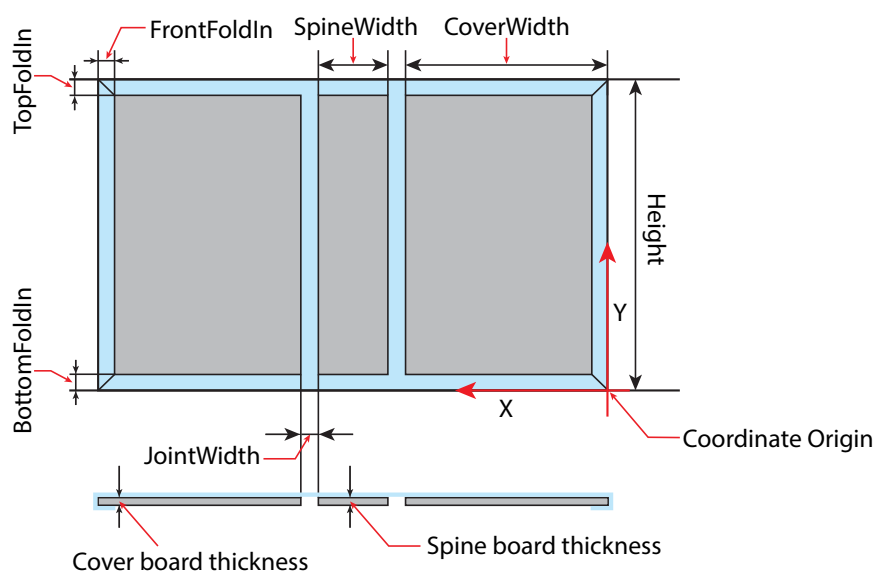
Input of Processes: **CaseMaking**

Table 6.23: CaseMakingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>BottomFoldIn</i> ?	float	Defines the width of the part of the cover material on the lower edge inside of the case. If <i>@BottomFoldIn</i> is not specified the value from <i>@TopFoldIn</i> SHALL be used.
<i>CornerType</i> ?	NMTOKEN	Method of wrapping the corners of the cover material around the corners of the board. Values include: LibraryCorner – The American Library Corner style.
<i>CoverWidth</i> ?	float	Width of the cover cardboard in points.
<i>FrontFoldIn</i> ?	float	Defines the width of the part of the cover material on the front edges inside of the case in points.
<i>Height</i> ?	float	Height of the book case, in points.
<i>JointWidth</i> ?	float	Width of the joint as seen when laying the cardboard on the cover material, in points.
<i>SpineWidth</i> ?	float	Width of the spine cardboard, in points.
<i>TopFoldIn</i> ?	float	Defines the width of the cover material on the top edge inside of the case, in points.
<i>Glue</i> ?	element	Details of the glue. Because the glue is applied to the whole back side of the cover material, <i>Glue</i> / <i>@AreaGlue</i> SHALL be set to "true".

The following figure shows the geometry of a book case. The thickness of the cover board and spine board are defined in the input **Component**(CoverBoard) and, optionally, the input **Component**(SpineBoard) of the **CaseMaking** process.

Figure 6-7: CaseMakingParams



6.13 CasingInParams

CasingInParams describes the settings of a **CasingIn** process. The geometry SHALL always be centered. See ▶ Figure 6-8: Parameters and coordinate system for CasingIn.

Resource Properties

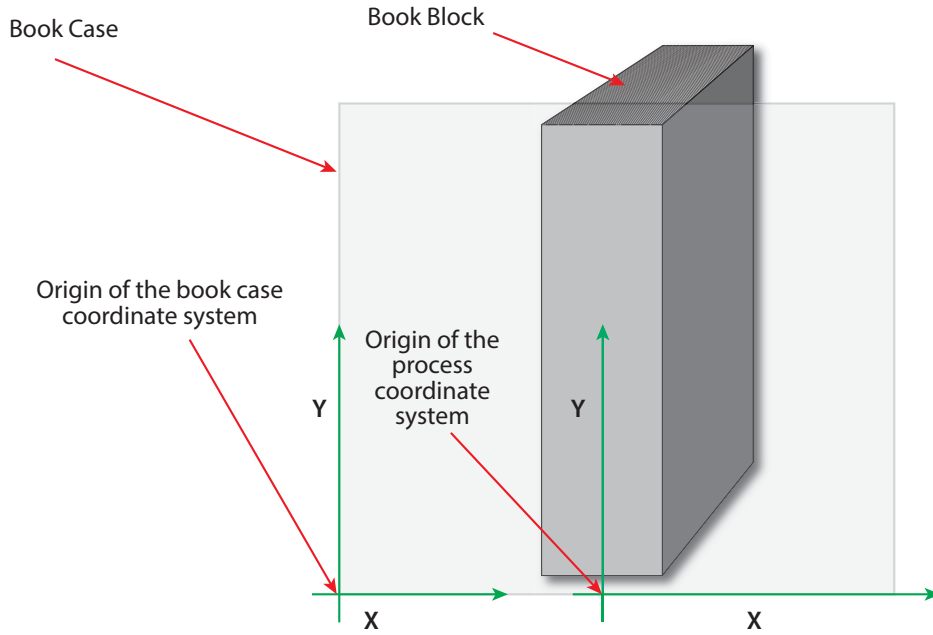
Intent Pairing: **BindingIntent**

Input of Processes: **CasingIn**

Table 6.24: CasingInParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>CaseRadius</i> ?	float	Inner radius of the case spine rounding. If not specified, no rounding of the case spine is performed.
<i>CoverWidth</i> ?	float	Width of the cover board. Note: Height and total case dimensions are specified in the Component (BookCase) of the CasingIn process. For details of @CoverWidth, see also ▶ Figure 6-7: CaseMakingParams.
<i>SpineWidth</i> ?	float	Width of the spine board. Note: Height and total case dimensions are specified in the Component (BookCase) of the CasingIn process. For details of @SpineWidth, see also ▶ Figure 6-7: CaseMakingParams.
<i>Glue</i> *	element	Properties of the glue to attach the case.

Figure 6-8: Parameters and coordinate system for CasingIn



6.14 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. 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**.

Resource/Part/@Separation SHALL be specified for each **ResourceSet**[@Name="Color"]/**Resource**. Individual colors SHALL be referenced by selecting a **Resource/Color** with a matching **../Resource/Part/@Separation**. **Resource/Part/@Separation** SHOULD match the PDL separation name and thus **Color/@ActualColorName** unless the separation name cannot be represented as an NMOKEN.

Resource Properties

Intent Pairing: **ColorIntent**

Input of Processes: **Any Process**

Table 6.25: Color Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ActualColorName</i> ?	string	Actual name of the color in the PDL. If not specified, the value from ../Part/@Separation SHALL be used. If @ActualColorName and ../Part/@Separation are different, the separation SHALL be identified by the value of ../Part/@Separation .
<i>CMYK</i> ?	CMYKColor	CMYK value of the 100% tint value of the colorant. @CMYK SHOULD be specified if known and @ColorType != "Transparent" and @ColorType != "DieLine".
<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.
<i>ColorBookEntry</i> ?	string	Definition of the Color within the standard specified by @ColorBook . This entry SHALL match the color book entry exactly as defined by the @ColorBook 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>ColorDetails</i> ?	string	A more specific, specialized or site-defined name for the color. If @ColorDetails is supplied, @ColorName SHOULD also be supplied.

Table 6.25: Color Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ColorName</i> ?	enumeration	Mapping to a color name. Allowed value is from: ▶ NamedColor.
<i>ColorType</i> ?	enumeration	A name that characterizes the colorant. Allowed values are: DieLine – Marks made with colorants of this type are ignored for trapping. Trapping processes need not generate a color plane for this colorant. "DieLine" can be used for auxiliary process separations. "DieLine" 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 @ColorType="DieLine" does not implicitly remove the "DieLine" separation from the final output. Normal – 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. Opaque – Marks covered by colorants of this type are ignored for trapping. "Opaque" can be used for metallic inks. Opaquelgnore – Marks made with colorants of this type and marks covered by colorants of this type are ignored for trapping. "Opaquelgnore" can be used for metallic inks. Primer – Colors with @ColorType="Primer" are used as background filler and SHALL be ignored when trapping. Transparent – 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.
<i>ColorTypeDetails</i> ?	string	Additional information about the color type. If @ColorType="DieLine" , this attribute SHOULD specify the type of die line, e.g. DDES-numbers. See ▶ Table 6.27 Diecutting Data (DDES3) for a list of DDES3 die line types.
<i>Density</i> ?	float	Density value of colorant (100% tint). Whereas @NeutralDensity 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.
<i>Gray</i> ?	float	Gray value of the 100% tint value of the colorant. @Gray SHALL be specified using 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>NeutralDensity</i> ?	float	A number in the range of 0.001 to 10 that represents the neutral density of the colorant, defined as $10 \cdot \log(1/Y)$. Y is the tristimulus value in CIEXYZ coordinates, normalized to 1.0.
<i>PrintingTechnology</i> ?	NMTOKEN	Printing technology of the press, press module or printer that this Color is intended for. Values include those from: ▶ Printing Technologies.
<i>PrintStandard</i> ?	string	Specifies the reference name of a characterization data set that this color is used in. See ▶ Appendix A.3.16 PrintStandard Characterization Data Sets for details. See also Part/@PrintCondition .
<i>RawName</i> ?	hexBinary	Representation of the original 8-bit byte stream of the ../Part/@Separation . Used to transport the original byte representation of a ../Part/@Separation 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> ?	RGBColor	sRGB value of the 100% tint value of the colorant. @sRGB SHOULD only be used for display purposes.
DeviceNColor *	element	Each DeviceNColor element defines the colorant in the DeviceN color space that is defined by DeviceNColor/@Name .

6.14.1 DeviceNColor

Table 6.26: DeviceNColor Element

NAME	DATA TYPE	DESCRIPTION
<i>ColorList</i>	FloatList	Value of the 100% tint value of the colorant in the <i>ColorantControl/DeviceNSpace</i> that is selected by <i>@Name</i> . Each index SHALL refer to the separation defined at the same position in <i>DeviceNSpace/@Separations</i> . A value of 0 SHALL specify no ink and a value of 1 SHALL specify full ink.
<i>Name</i>	NMTOKEN	<i>@Name</i> of the matching <i>DeviceNSpace</i> . Exactly one <i>ColorantControl/DeviceNSpace/@Name</i> SHALL match <i>@Name</i> .

6.14.2 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.

Table 6.27: Diecutting Data (DDES3)

DDES3 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.
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.

6.15 ColorantControl

ColorantControl defines how color separations of PDL or raster data SHALL be output on a target device or file. 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

Intent Pairing: **ColorIntent**

Input of Processes **ColorCorrection, ColorSpaceConversion, ConventionalPrinting, DigitalPrinting, ImageSetting, Interpreting, PreviewGeneration, Separation, Stripping, Trapping**

Output of Processes: **ColorSpaceConversion**

Table 6.28: ColorantControl Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ColorantConvertProcess</i> ?	NMTOKENS	List of colors that SHALL be converted to process colors. If not specified then all colors that are not listed in <i>@ColorantParams</i> SHALL be converted. An application MAY issue a warning for all PDL colors that are not in either of <i>@ColorantParams</i> or <i>@ColorantConvertProcess</i> .
<i>ColorantOrder</i> ?	NMTOKENS	The ordering of the colorant identifiers to be processed. All of the colorants named SHALL occur in the <i>@ColorantParams</i> list. If present, then only the colorants specified by <i>@ColorantOrder</i> SHALL be output. Colorants listed in the <i>@ColorantParams</i> , but not listed in <i>@ColorantOrder</i> , 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 <i>@ColorantParams</i> SHALL be output.
<i>ColorantParams</i> ?	NMTOKENS	<i>@ColorantParams</i> defines all the colorant identifiers that are expected to be available on the printing device. Colorant identifiers found in the PDL that are not listed in <i>@ColorantParams</i> SHALL be implemented through their <i>@ProcessColorModel</i> equivalents. (See ColorSpaceConversion process.) The colorants implied by the value of <i>@ProcessColorModel</i> SHALL be specified in this list. Note: The spot colors defined in ColorIntent/SurfaceColor/@ColorsUsed will in general be mapped to <i>@ColorantParams</i> for each spot color to be used as part of any product intent to process conversion.
<i>MappingSelection</i> ?	enumeration	<i>@MappingSelection</i> specifies how a combination of process colorant values SHALL be obtained for any spot color when the separation spot colorant itself is not available. Allowed value is from: ▶ MappingSelection.
<i>ProcessColorModel</i> ?	enumeration	Specifies the model to be used for rendering the colorants defined in color spaces into process colorants. Allowed values are: DeviceCMY – Process colors SHALL be Cyan Magenta and Yellow. DeviceCMYK – Process colors SHALL be Cyan Magenta Yellow and Black. DeviceGray – Process color SHALL be Black. DeviceN – The specific DeviceN color space to operate on is defined in the <i>DeviceNSpace</i> resource. If this value is specified then <i>DeviceNSpace</i> SHALL also be present and the process colors SHALL be specified in <i>DeviceNSpace/@Separations</i> . DeviceRGB – Process colors SHALL be Red Green and Blue. None – No colorants other than those specified in <i>@ColorantParams</i> SHALL be output. Note: The separations implied by <i>@ProcessColorModel</i> SHALL be explicitly specified in <i>@ColorantParams</i> .
<i>ColorantAlias</i> *	element	Each <i>ColorantAlias</i> is used to map a color name from the PDL to a separation.

Table 6.28: ColorantControl Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>DeviceNSpace</i> ?	element	<i>DeviceNSpace</i> defines the colorants that make up a DeviceN color space. <i>DeviceNSpace</i> SHALL be present if the <i>@ProcessColorModel</i> value is "DeviceN".

6.15.1 ColorantAlias

ColorantAlias is an element that specifies a replacement colorant name string to be used instead of one or more named colorant strings.

Table 6.29: 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 separation identifier that SHALL be selected for the colorant <i>@ColorantName</i> .

Example 6.6: ColorantAlias/@RawName

The following example shows how the two colorants "Grün" and "grün" are mapped to the separation identifier "Spot1" which is mapped to the actual separation "Green".

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="RawName" Types="ColorSpaceConversion">
  <ResourceSet Name="ColorantControl" Usage="Input">
    <Resource>
      <ColorantControl>
        <ColorantAlias ColorantName="Grün" RawName="4772FC6E" ReplacementColorantName="Spot1"/>
        <ColorantAlias ColorantName="grün" RawName="6772FC6E" ReplacementColorantName="Spot1"/>
      </ColorantControl>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Color" Usage="Input">
    <Resource>
      <Part Separation="Spot1"/>
      <Color ActualColorName="Green" RawName="477265656E" sRGB="0 1 0"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

The following table describes which separations are output for various values of *@ProcessColorModel*, *@ColorantOrder*, and *@ColorantParams*. 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 6.30: Sample output for different values of ProcessColorModel, ColorantParams and ColorantOrder (Sheet 1 of 2)

PROCESSCOLORMODEL	COLORANTPARAMS	COLORANTORDER	COLORANTS NOT SHOWN IN THE OUTPUT
DeviceCMYK	Cyan Magenta Yellow Black	Cyan Magenta	Yellow Black

Table 6.30: Sample output for different values of ProcessColorModel, ColorantParams and ColorantOrder (Sheet 2 of 2)

PROCESSCOLORMODEL	COLORANTPARAMS	COLORANTORDER	COLORANTS NOT SHOWN IN THE OUTPUT
DeviceCMYK	Cyan Magenta Yellow Black Spot1 Spot2	Cyan Magenta Yellow Black Spot2	Spot1
DeviceGray	Black Spot1 Spot2	Black Spot2	Spot1
DeviceN (with example N = 2 colorants as identified in DeviceNSpace)	DeviceN1 DeviceN2 Spot1 Spot2	Spot2 DeviceN1 DeviceN2	Spot1

6.15.2 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 6.31: DeviceNSpace Element

NAME	DATA TYPE	DESCRIPTION
Name	NMTOKEN	Name of the DeviceNSpace.
Separations	NMTOKENS	Ordered list of colorant identifiers that define the DeviceN color space. Additional details of the colorants SHOULD be provided in ResourceSet[@Name="Color"].

6.16 ColorCorrectionParams

ColorCorrectionParams provides the information needed to algorithmically correct colors on some PDL pages or content elements such as image, graphics or formatted text.

Resource Properties

Intent Pairing: ColorIntent

Input of Processes: ColorCorrection

Table 6.32: 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.

6.16.1 ColorCorrectionOp

If present, the following attributes SHALL be applied at a point where an abstract profile would be applied following any abstract profiles used in the order: @AdjustLightness, @AdjustContrast, @AdjustSaturation, @AdjustHue.

Table 6.33: ColorCorrectionOp Element

NAME	DATA TYPE	DESCRIPTION
<i>AdjustContrast</i> ?	float	Specifies the L*a*b* contrast adjustment in the range -100 to + 100. <i>@AdjustContrast</i> SHALL scale the L* channel about mid-scale (where L* = 50). $L_{Adjust} = 50 + (L - 50) * (@AdjustContrast / 100 + 1)$ Note: 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.
<i>AdjustHue</i> ?	float	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. $a_{Adjust} = a * \cos(@AdjustHue) - b * \sin(@AdjustHue)$ $b_{Adjust} = a * \sin(@AdjustHue) + b * \cos(@AdjustHue)$
<i>AdjustLightness</i> ?	float	Specifies the decrease or increase of the L*a*b* lightness in the range -100 to + 100. <i>@AdjustLightness</i> SHALL offset the L* channel. $L_{Adjust} += @AdjustLightness.$ Note: Increasing the lightness value causes the output to appear lighter and decreasing the lightness value causes the output to appear darker.
<i>AdjustSaturation</i> ?	float	Specifies the increase or decrease of the L*a*b* color saturation in the range -100 to +100. <i>@AdjustSaturation</i> SHALL scale the a* and b* channels around zero. $a_{Adjust} *= (@AdjustSaturation / 100 + 1)$ $b_{Adjust} *= (@AdjustSaturation / 100 + 1)$ Note: 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.
<i>SourceObjects</i> ?	enumerations	Identifies which class(es) of incoming graphical objects SHALL be operated on. If <i>@SourceObjects</i> is not specified then ColorCorrectionOp SHALL apply to all object classes. Allowed values are from: ▶ SourceObjects.

6.17 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 define 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

Intent Pairing: **ColorIntent**

Input of Processes: **ColorSpaceConversion**

Table 6.34: ColorSpaceConversionParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ICCProfileUsage</i> ?	enumeration	<i>@ICCProfileUsage</i> specifies where to obtain either the destination profile or device link transform that SHALL be applied. 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 of the source object from the source space directly to the destination. Note: PDF/X workflows assume that <i>@ICCProfileUsage</i> ="UsePDL". Allowed values are: UsePDL – If present, the embedded target profile SHALL be used. UseSupplied – The embedded target profile SHALL NOT be used.

Table 6.34: ColorSpaceConversionParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ColorSpaceConversionOp</i> *	element	List of <i>ColorSpaceConversionOp</i> 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 <i>ColorSpaceConversionOp</i> elements is significant, and when multiple elements apply to the same object, they are applied in that XML order. A <i>ColorSpaceConversionOp</i> can modify the characteristics of an object such that its selection criteria is also modified. Thus, if two <i>ColorSpaceConversionOp</i> 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 <i>ColorSpaceConversionOp</i> SHALL NOT be applied to that object.
<i>FileSpec</i> ?	element	A <i>FileSpec</i> element pointing to an ICC profile that describes the characterization of the final output target device.

6.17.1 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].

Table 6.35: ColorSpaceConversionOp Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Operation</i>	enumeration	@ <i>Operation</i> specifies the task that this <i>ColorSpaceConversionOp</i> defines. Allowed values are: <i>Convert</i> – Selected graphical elements SHALL be transformed into the target color space. <i>Tag</i> – Selected graphical elements SHALL be tagged with the source color space defined in <i>FileSpec(SourceProfile)</i> . <i>Untag</i> – The source profile SHALL be removed from all selected graphical elements.
<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	Identifies the rendering intent to be applied when rendering the objects selected by this <i>ColorSpaceConversionOp</i> . Allowed value is from: ▶ <i>RenderingIntent</i> .
<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> . Gray values that exceed the @ <i>RGBGray2BlackThreshold</i> SHALL NOT be converted. @ <i>RGBGray2Black</i> and @ <i>RGBGray2BlackThreshold</i> are used by the <i>ColorSpaceConversion</i> process in determining how to allocate RGB values to the black (K) channel.
<i>RGBGray2BlackThreshold</i> ?	float	When @ <i>RGBGray2Black</i> is "true", @ <i>RGBGray2BlackThreshold</i> SHALL be a value between "0.0" and "1.0" that specifies the threshold value above which the device SHALL NOT convert gray (R = G = B) to black (K only). Thus a value of "0.0" will convert only R = G = B = 0 (black) to K, whereas a value of "1.0" will convert all values of R = G = B to K.
<i>Separations</i> ?	NMTOKENS	List of separation identifiers that specify on which separation(s) to operate when @ <i>SourceCS</i> ="Separation". Additional details of the colorants SHOULD be provided in <i>ResourceSet</i> [@Name="Color"].

Table 6.35: ColorSpaceConversionOp Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>SourceCS</i> ?	enumeration	Identifies which of the incoming color spaces SHALL be operated on. If @ <i>SourceCS</i> is not specified then <i>ColorSpaceConversionOp</i> SHALL apply to all color spaces. Allowed value is from: ▶ SourceColorSpace. Note: See ▶ Appendix A.2.39.1 Source color space mapping.
<i>SourceObjects</i> ?	enumerations	List of object classes that identifies which incoming graphical objects SHALL be operated on. If @ <i>SourceObjects</i> is not specified then <i>ColorSpaceConversionOp</i> SHALL apply to all object classes. Allowed values are from: ▶ SourceObjects.
<i>SourceRenderingIntent</i> ?	enumeration	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. If not specified then the value from @ <i>RenderingIntent</i> SHOULD be used. Allowed value is from: ▶ RenderingIntent. Note: The @ <i>SourceRenderingIntent</i> will pertain to the source profile used in a particular <i>ColorSpaceConversion</i> process (e.g., sources can be the native original color space, an intermediate working color space or a reference output simulation color space).
<i>FileSpec</i> (<i>DeviceLinkProfile</i>) ?	element	<i>FileSpec</i> (<i>DeviceLinkProfile</i>) specifies an ICC profile file that contains a device link transform. See ▶ [ICC.1].
<i>FileSpec</i> (<i>PDLSourceProfile</i>) ?	element	<i>FileSpec</i> (<i>PDLSourceProfile</i>) specifies an ICC profile that describes a profiled source color space that this <i>ColorSpaceConversionOp</i> SHALL operate on. When present, only objects that are tagged with this profile SHALL be selected.
<i>FileSpec</i> (<i>SourceProfile</i>) ?	element	<i>FileSpec</i> (<i>SourceProfile</i>) specifies an ICC profile that SHALL be used as the profile for the source object's color space during a "Convert" or "Tag" operation, as specified by @ <i>Operation</i> . <i>FileSpec</i> (<i>SourceProfile</i>) SHALL be present if @ <i>Operation</i> ="Tag". <i>FileSpec</i> (<i>SourceProfile</i>) SHALL NOT be present if @ <i>Operation</i> ="Untag".
<i>ScreenSelector</i> ?	element	If specified, only objects that match the screening properties defined in <i>ScreenSelector</i> SHALL be operated on.

6.18 Component

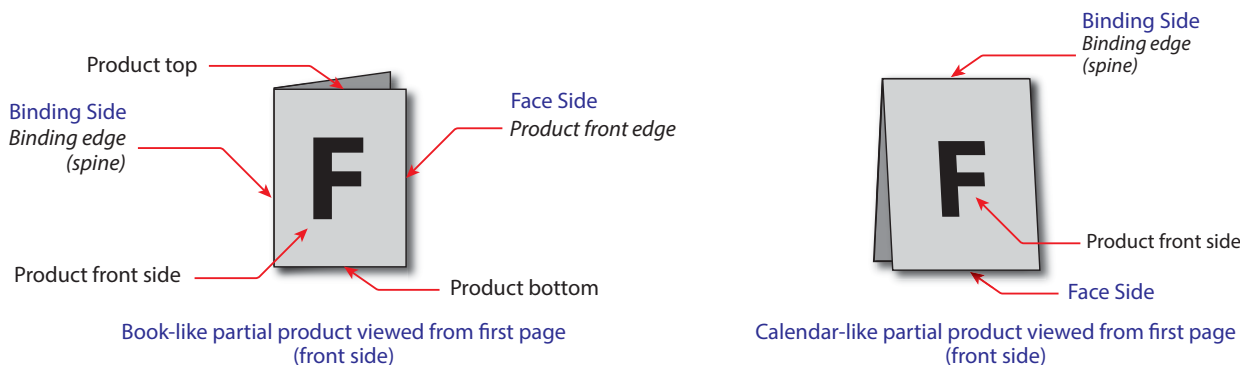
Component is used to describe the various versions of semi-finished goods in the press and postpress area, such as a pile of folded sheets that have been collected and are then to be joined and trimmed. Nearly every postpress process has a **Component** resource as an input as well as an output.

Glossary – Component

The descriptions of **Component** 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 6-9: Component – terms and definitions



Resource Properties

Resource referenced by: **Bundle/BundleItem, FeedingParams/Feeder, FeedingParams/CollatingItem**

Input of Processes: **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, Stitching, Strapping, ThreadSealing, ThreadSewing, Trimming, WebInlineFinishing, Winding, Wrapping**

Output of Processes: **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, Stitching, Strapping, ThreadSealing, ThreadSewing, Trimming, WebInlineFinishing, Winding, Wrapping**

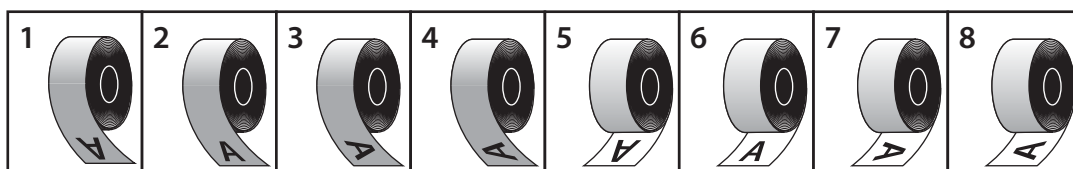
Table 6.36: Component Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Automation</i> ?	enumeration	Identifies dynamic and static components. When a Component is referenced from a binding process, @Automation modifies the scope of the Component that SHALL be bound. If @Automation="Static" , the individual Component elements that SHALL be bound are one instance of the referenced Component . If @Automation="Dynamic" , the individual Component elements that SHALL be bound are identified by the Resource/Part of this Component . Generating Resource/Part in automated imposition is defined in detail in: ▶ Section 5.4.8.1 Execution Model for Automated Imposition. If @Automation="Dynamic" and ResourceSet/Dependent/@PipeID is also present, details MAY be specified in XJMF pipe messages. See ▶ Section 9.3.5.1 Dynamic Pipes. If an IdentificationField/MetadataMap element is present, the details SHALL be controlled by the barcode that is represented by IdentificationField/MetadataMap . Allowed value is from: ▶ Automation.
<i>CartonTopFlaps</i> ?	XYPair	Size (F1,F2) (see ▶ Figure 6-4: Box packing) of the two top flaps of a carton or box. SHALL NOT be specified unless @ProductType="Carton" or @ProductType="Box" .
<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>ContentRefs</i> ?	IDREFS	Reference to Content that provides metadata related to this Component .

Table 6.36: Component Resource (Sheet 2 of 2)

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 Component . If not specified, a portrait orientation (Y > X) is assumed. If an unfolded Component references Media , then the Z-value of <i>@Dimensions</i> SHOULD correspond to the value of Media / <i>@Thickness</i> transformed from microns to points. Note: It is crucial for enabling postpress to specify <i>@Dimensions</i> unless they really are unknown.
<i>MaxHeat</i> ?	float	Maximum temperature the Component can resist.
<i>MediaRef</i> ?	IDREF	Reference to the Media for this Component .
<i>Overfold</i> ?	float	Expansion of the overfold of a Component . This attribute is needed for Inserting or other postpress processes.
<i>OverfoldSide</i> ?	enumeration	Specifies the longer side of a folded component. Allowed value is from: ▶ Side.
<i>ProductType</i> ?	NMTOKEN	Type of product that this component specifies. Values include those from: ▶ Product Types.
<i>ProductTypeDetails</i> ?	string	<i>@ProductTypeDetails</i> specifies additional details of the product or product part that may be site specific and may be human readable. <i>@ProductType</i> SHOULD be specified if <i>@ProductTypeDetails</i> is supplied. 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>ReaderPageCount</i> ?	integer	Total amount of individual reader pages that this Component contains.
<i>SurfaceCount</i> ?	integer	Total amount of individual surfaces that this Component contains. Note: A sheet always has two surfaces regardless of the number of images or reader pages. In case of homogeneous Component elements, <i>@SurfaceCount</i> refers to surfaces with a size of Component / <i>@Dimensions</i> .
<i>WindingResult</i> ?	integer	Orientation of the finished product on the roll. For an image, see ▶ Figure 6-10: Orientation of the finished product on the roll. The integers in the figure correspond to values specified by the labeling trade association, refer to ▶ [FINAT]. Note: The orientation and number of windings in a Winding process are modified based on the value of <i>@WindingResult</i> .
<i>IdentificationField</i> *	element	IdentificationField associates bar codes or labels with this Component .

Figure 6-10: Orientation of the finished product on the roll



6.19 Contact

Contact describes a person or a role within an organization. It MAY include an address and communication channels. The *@ExternalID* attribute in the parent **Resource** SHALL be unique within the company.

Resource Properties

Resource referenced by: [ApprovalParams/ApprovalPerson](#), [Content/ContentMetadata](#)Input of Processes: [Any Process](#)

Table 6.37: Contact Resource

NAME	DATA TYPE	DESCRIPTION
ContactTypeDetails ?	string	@ContactTypeDetails specifies the details of the contact's role or roles. If Part/@ContactType="Employee" , then @ContactTypeDetails SHALL define the list of roles that the employee fills. Values include: Apprentice – Employee that is in training (“Auszubildender” / “Auszubildende” / “Stift” in German). Assistant – Assistant operator. Craftsman – Trained employee (“Geselle” / “Facharbeiter” in German). CSR – Customer Service Representative. Manager – Manager. Master – Highly trained employee (“Meister” in German). Operator – Operator. ShiftLeader – The leader of the shift. StandBy – Employee who is allocated to a specific task on demand.
CostCenterID ?	NMTOKEN	Identifier of the cost center that this Contact belongs to if Part/@ContactType="Employee" .
UserID ?	string	User ID of user, as specified when logging into the operating system or into the submitting application.
Address ?	element	Element describing the address.
ComChannel *	element	Communication channels such as phone number or email of the contact.
Company ?	element	Company that this Contact is associated with.
Person ?	element	Name of the contact person.

6.19.1 ComChannel

A communication channel to a [Contact](#) such as an email address, phone number or fax number.

Table 6.38: ComChannel Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
ChannelType	NMTOKEN	Type of the communication channel. Values include: Email – Email address. Fax – Fax machine. JMF – XJMF messaging channel. Mobile – Mobile phone. Phone – Telephone number. This SHOULD be restricted to land line phones. WWW – WWW home page or form.
ChannelUsage ?	NMTOKENS	Communication channel usage. Values include: Business – Business purpose usage (e.g., office phone number, fax). DayTime – Office hours in the time zone of the Contact . NightTime – Out-of-office hours in the time zone of the Contact . Private – Private purpose usage (e.g., private phone number, fax, email). WeekEnd – Out-of-office days in the time zone of the Contact .
DescriptiveName ?	string	Human readable representation of ComChannel .

Table 6.38: ComChannel Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Locator</i>	string	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 ▶ [RFC6068] for “mailto” URLs, ▶ [RFC3966] for “tel” URLs and ▶ [RFC3986] for URLs in general. For example, use "mailto:a@b.com" instead of "a@b.com" if <i>@ChannelType</i> ="Email" "tel:+49-69-92058800" if <i>@ChannelType</i> ="Phone" "tel:+49.6151.155.299" if <i>@ChannelType</i> ="Fax".

6.19.2 Company

Company defines the organization name and organizational units (ORG) of the organizational properties defined in ▶ [vCard]. *@CompanyID* SHALL be globally unique across all companies.

Table 6.39: Company Element

NAME	DATA TYPE	DESCRIPTION
<i>CompanyID</i> ?	NMTOKEN	An ID of the company. The <i>@CompanyID</i> attribute for each specified company SHALL be globally unique across all companies.
<i>DescriptiveName</i> ?	string	Human readable representation of Company including any OrganizationalUnit elements.
<i>OrganizationName</i>	string	Name of the organization or company (vCard: ORG:orgnam, e.g. “ABC Inc.”).
OrganizationalUnit *	element	Describes one or more organizational units (vCard: ORG:orgunit), e.g. if two elements are present: 1. “North American Division” and 2. “Marketing”.

6.19.3 OrganizationalUnit

Table 6.40: OrganizationalUnit Element

NAME	DATA TYPE	DESCRIPTION
	text	Description of one organizational unit.

6.19.4 Person

Person provides detailed information about a person. The structure of **Person** is derived from the vCard format, see ▶ [vCard]. The corresponding XML types of the vCard are quoted in the description field of the table below.

Table 6.41: Person Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>AdditionalNames</i> ?	string	Additional names of the contact person. See ▶ [vCard]N:other.
<i>DescriptiveName</i> ?	string	Human readable representation of Person . <i>@DescriptiveName</i> SHOULD be a combination of the name related attributes of Person . Note: The sequence of individual name parts is language dependent.
<i>FamilyName</i> ?	string	The family name of the contact person. See ▶ [vCard]N:family.
<i>FirstName</i> ?	string	The first name of the contact person. See ▶ [vCard]N:given.
<i>JobTitle</i> ?	string	Job function of the person in the company or organization. See ▶ [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. See ▶ [vCard]N:prefix.
<i>NameSuffix</i> ?	string	Suffix of the name. See ▶ [vCard]N:suffix.

Table 6.41: Person Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>PhoneticFirstName</i> ?	string	Alternative spelling of a first name, used to assist with pronunciation, e.g. for use with Kanji (Japanese) names). See ▶ [vCard]X-PHONETIC-FIRST-NAME.
<i>PhoneticLastName</i> ?	string	Alternative spelling of a last name, used to assist with pronunciation, e.g. for use with Kanji (Japanese) names). See ▶ [vCard]X-PHONETIC-LAST-NAME

6.20 Content

Resource Properties

Resource referenced by: [RunList](#)

Content defines the additional metadata of individual graphic elements.

Table 6.42: Content Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>BinderySignatureIDs</i> ?	NMTOKENS	If specified, @ <i>BinderySignatureIDs</i> SHALL list the <i>BinderySignature</i> / <i>@BinderySignatureID</i> of all bindery signatures that this Content applies to.
<i>ContentStatus</i> ?	NMTOKENS	Status of a single Content element. Values include those from: ▶ Table 6.43 ContentStatus Attribute Values.
<i>ContentType</i> ?	NMTOKEN	Type of content. Values include those from: ▶ Content Types.
<i>HasBleeds</i> ?	boolean	If "true", the Content has bleeds.
<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>Separations</i> ?	NMTOKENS	List of separation identifiers that are present in the content. Additional details of the colorants MAY be provided in <i>ResourceSet</i> [<i>@Name="Color"</i>].
<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>BarcodeProductionParameters</i> ?	element	Description of the specific parameters for barcode production.
<i>ContentMetadata</i> ?	element	Container for document related metadata such as ISBN, Author etc.
<i>FileSpec</i> *	element	Reference to dependent references such as fonts, external images, etc. Each <i>FileSpec</i> / <i>@ResourceUsage</i> SHOULD have one of the following values: Font – The file references a font. Image – The file references image data. PDL – The file references page definition language data such as PDF.
<i>ImageCompression</i> ?	element	Specification of the image compression properties.
<i>OCGControl</i> *	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.

Table 6.42: Content Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
PositionObj *	element	Definition of the size and positioning of any child Content elements.
ScreenSelector ?	element	Specification of the screening properties of the Content .

Table 6.43: ContentStatus Attribute Values

VALUE	DESCRIPTION
Accepted	The receiver acknowledged that the files are accessible for their destination.
Approved	Planned proofs have been approved.
BindingCompleted	All binding worksteps including packing of the job have been completed. Postpress worksteps are defined according to ▶ Section 5.6 Postpress Processes.
BindingInProgress	At least one of the binding worksteps of the job is in progress.
Deleted	Specifies that the content was deleted.
DigitalArtArrived	Digital content has been received.
PostPressCompleted	All postpress worksteps including packing of the job have been completed. Postpress worksteps are defined according to ▶ Section 5.6 Postpress Processes.
PostPressInProgress	At least one of the postpress worksteps of the job is in progress.
PrePressCompleted	All prepress worksteps of the job have been completed. Prepress worksteps are defined according to ▶ Section 5.4 Prepress Processes. In conventional prepress, this is the case when all plates have been made.
PrePressInProgress	At least one of the prepress worksteps of the job is in progress.
PressCompleted	All press worksteps of the job have been completed. Press worksteps are defined according to ▶ Section 5.5 Press Processes.
PressInProgress	At least one of the press worksteps of the job is in progress.
Proofed	Planned proofs have been made.
ProofSent	Planned proofs sent to customer.
ShippingCompleted	Final product was delivered to the customer or distributors.
ShippingInProgress	Final product is being shipped.
SurfaceAssigned	Pages have been assigned to their respective imposition layouts.

6.20.1 BarcodeProductionParams

[BarcodeProductionParams](#) describes the specific parameters for barcode production.

Table 6.44: BarcodeProductionParams Element

NAME	DATA TYPE	DESCRIPTION
BarcodeReproParams ?	element	Description of the formatting and reproduction parameters for barcode production.
IdentificationField ?	element	Description of the barcode metadata.

6.20.2 ContentMetadata

[ContentMetadata](#) is a container for metadata pertaining to this [Content](#) resource.

Table 6.45: ContentMetadata Element

NAME	DATA TYPE	DESCRIPTION
<i>ContactRefs</i> ?	IDREFS	@ <i>ContactRefs</i> SHALL reference the <i>Contacts</i> that represent the various contact types that are responsible for this content.
<i>ISBN</i> ?	NMTOKEN	An International Standard Book Number, that allows for both 10 and 13 digit values, see ▶ [ISO2108:2017].
<i>Title</i> ?	NMTOKEN	The title of the content.
<i>Comment</i> ?	element	If required, an abstract MAY be specified in <i>Comment</i> [@Name="Abstract"].
<i>GeneralID</i> *	element	Additional metadata MAY be defined by adding <i>GeneralID</i> elements.

6.20.3 PositionObj

PositionObj describes the size and position of the *Content*.

Table 6.46: PositionObj Element

NAME	DATA TYPE	DESCRIPTION
<i>Anchor</i> ?	enumeration	Specifies the origin (0,0) of the coordinate system in the unrotated <i>Content</i> . Allowed value is from: ▶ <i>Anchor</i> .
<i>CTM</i> ?	matrix	Specifies the transformation matrix of the origin of <i>Content</i> as specified by @ <i>Anchor</i> . Note: This is not necessarily the actual CTM that will position a given <i>Content</i> . The actual CTM SHALL be recalculated based on the values of @ <i>Anchor</i> and @ <i>Size</i> .
<i>PageRange</i> ?	IntegerRange	Reader page index in the <i>Content</i> referenced by <i>RefAnchor</i> .
<i>PositionPolicy</i> ?	enumeration	Specifies the level of freedom when applying the values specified in <i>PositionObj</i> . Allowed value is from: ▶ <i>PositionPolicy</i> .
<i>RelativeSize</i> ?	XYPair	Specifies the size of the unrotated and unscaled object, relative to the parent specified in <i>RefAnchor</i> .
<i>RotationPolicy</i> ?	enumeration	Specifies the level of freedom when applying the values specified in <i>PositionObj</i> . Allowed value is from: ▶ <i>PositionPolicy</i> .
<i>Size</i> ?	XYPair	Specifies the size of the unrotated and unscaled object, in points.
<i>SizePolicy</i> ?	enumeration	Specifies the level of freedom when applying the values specified in <i>PositionObj</i> . Allowed value is from: ▶ <i>PositionPolicy</i> .
<i>RefAnchor</i> ?	element	Reference to a <i>Content</i> that this <i>Content</i> is positioned relative to. If <i>RefAnchor</i> is not specified, <i>PositionObj</i> refers to the lower left of the first page specified in @ <i>PageRange</i> .

6.21 ConventionalPrintingParams

ConventionalPrintingParams defines the device specific setup of the *ConventionalPrinting* process.

Resource Properties

Input of Processes: **ConventionalPrinting**

Table 6.47: ConventionalPrintingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>Drying</i> ?	enumeration	The way in which ink is dried after a print run. Allowed value is from: ▶ Drying.
<i>FirstSurface</i> ?	enumeration	Printing order of the surfaces on the sheet. Allowed value is from: ▶ Side.
<i>FountainSolution</i> ?	enumeration	State of the fountain solution module in the printing units. Allowed values are: On Off
<i>ModuleDrying</i> ?	enumeration	The way in which ink is dried in individual modules. Allowed value is from: ▶ Drying.
<i>Powder</i> ?	float	Quantity of powder in percent.
<i>SheetLay</i> ?	enumeration	Lay of input media. Reference edge of where paper is placed in a feeder. Allowed value is from: ▶ SheetLay.
<i>Speed</i> ?	float	Maximum print speed in the units as specified by ResourceSet [@Name="Component" AND @Usage="Output"]/@Unit per hour. If not specified then the press speed need not be reduced.
<i>WorkStyle</i> ?	enumeration	The direction in which to turn the press sheet. Allowed value is from: ▶ WorkStyle.

6.22 CoverApplicationParams

CoverApplicationParams define the parameters for applying a cover to a book block.

Resource Properties

Intent Pairing: **BindingIntent**

Input of Processes: **CoverApplication**

Table 6.48: CoverApplicationParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>Glue</i> *	element	Describes where and how to apply glue to the book block.
<i>Score</i> *	element	Describes where and how to score the cover. The sequence of Score elements SHALL specify the sequence in which the tool is applied.

6.22.1 Score

Table 6.49: Score Element

NAME	DATA TYPE	DESCRIPTION
<i>Offset</i>	float	Position of scoring given in the operation coordinate system.
<i>Side</i>	enumeration	Specifies the side from which the scoring tool works. Allowed values are: FromInside FromOutside

Figure 6-11: Parameters and coordinate system for cover application

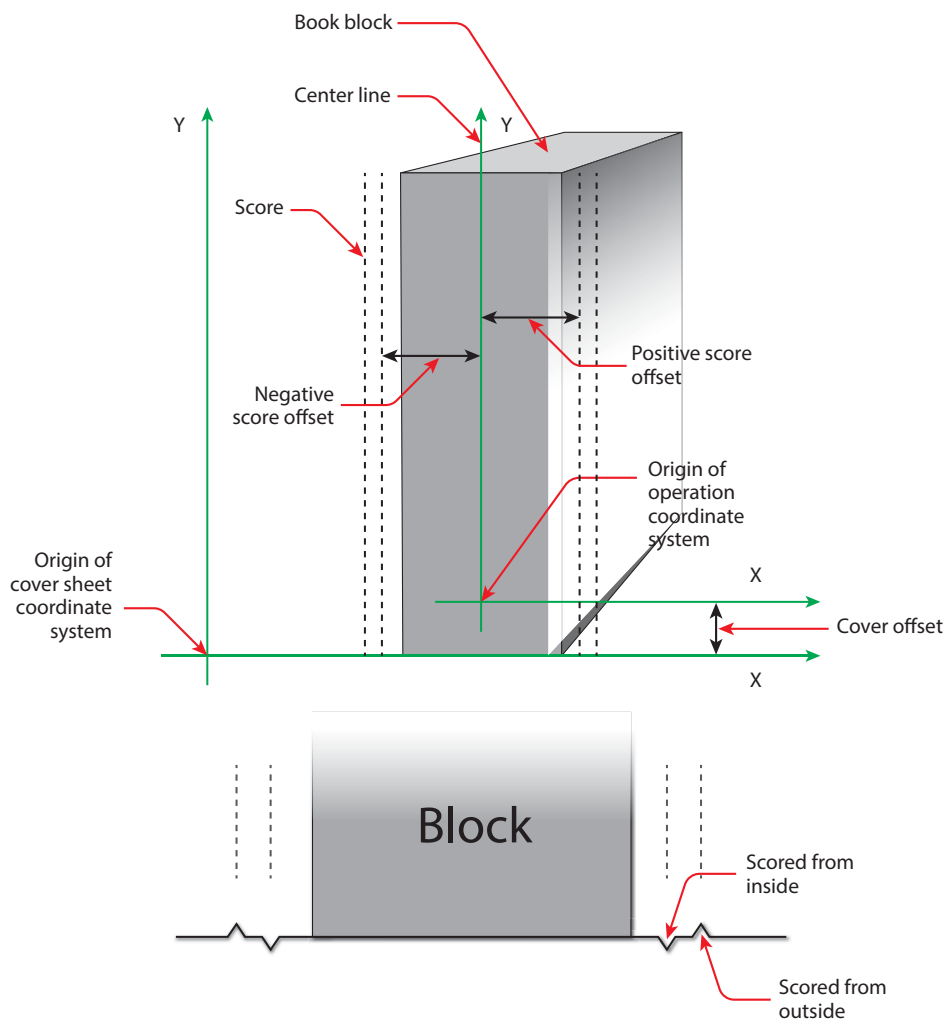
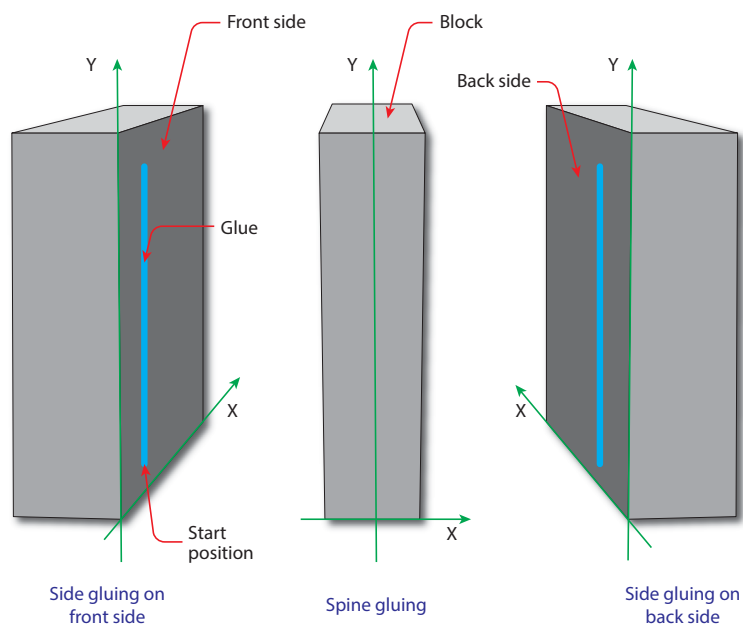


Figure 6-12: Parameters and coordinate system for glue application



6.23 CreasingParams

CreasingParams define the parameters for creasing or grooving a sheet.

RESOURCES

Resource Properties

Intent Pairing: [FoldingIntent](#)

Input of Processes: [Creasing](#)

Table 6.50: *CreasingParams* Resource

NAME	DATA TYPE	DESCRIPTION
Crease +	element	Each Crease element defines one crease line.

6.24 CustomerInfo

The [CustomerInfo](#) resource contains information about the customer who orders the job.

Note: Additional details about the customer can be found in [Contact](#) with [Part](#)/[@ContactType](#)="Customer".

Resource Properties

Input of Processes: [Any Process](#)

Table 6.51: *CustomerInfo* Resource

NAME	DATA TYPE	DESCRIPTION
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.

6.25 CuttingParams

[CuttingParams](#) 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

Intent Pairing: [FoldingIntent](#), [ShapeCuttingIntent](#)

Input of Processes: [Cutting](#)

Table 6.52: *CuttingParams* Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
SheetLay ?	enumeration	Lay of the input Component . Allowed value is from: ▶ SheetLay . Note: @SheetLay does not modify the coordinate references of the Cutting process.
Cut *	element	Cut elements describe an individual cut. The cuts shall be performed in the same sequence as they occur in this CuttingParams . 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 Component . CutBlock elements SHALL NOT be specified if Cut elements are specified. CuttingParams SHALL NOT be partitioned by @BlockName values that match CutBlock / @BlockName . Note: The partitioning structure of CuttingParams describes the original source cut block prior to cutting. The CutBlock elements describe the resulting cut blocks. This allows the description of nested cut blocks by providing multiple Cutting processes.

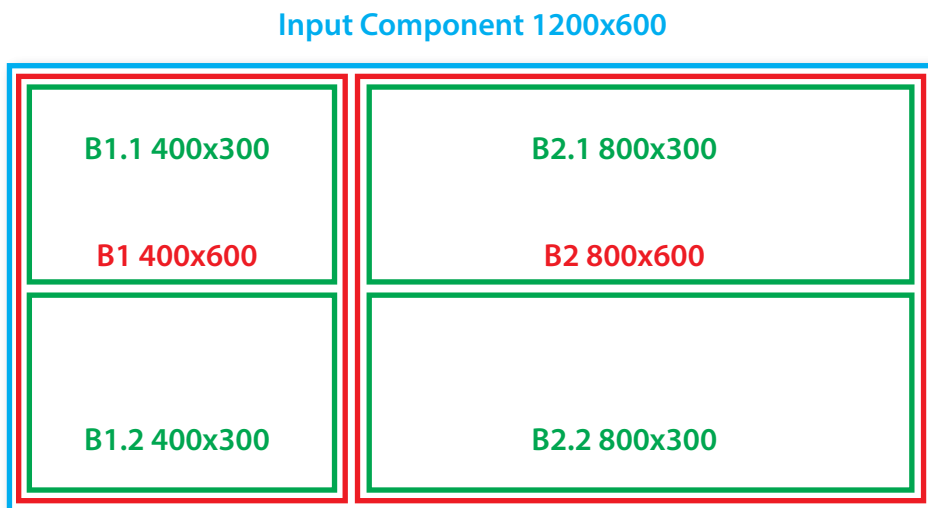
Table 6.52: CuttingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
FileSpec (CIP3) ?	element	Reference to a CIP3 file that contains cutting instructions in the ▶ [CIP3 - PPF] format. If FileSpec (CIP3) is specified, Cut and CutBlock SHALL NOT be present.

Example 6.7: Nested Cut Blocks

The following figure illustrates the nested cut blocks example.

Figure 6-13: Nested cut blocks



RESOURCES

```

<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="Bundle" Types="Cutting Cutting">
  <ResourceSet CombinedProcessIndex="0" Name="CuttingParams" Usage="Input">
    <Resource>
      <CuttingParams>
        <CutBlock BlockName="B1" Box="0 0 400 600"/>
        <CutBlock BlockName="B2" Box="400 0 1200 600"/>
      </CuttingParams>
    </Resource>
  </ResourceSet>
  <ResourceSet CombinedProcessIndex="1" Name="CuttingParams" Usage="Input">
    <Resource>
      <Part BlockName="B1"/>
      <CuttingParams>
        <CutBlock BlockName="B1.1" Box="0 0 400 300"/>
        <CutBlock BlockName="B1.2" Box="0 300 400 600"/>
      </CuttingParams>
    </Resource>
    <Resource>
      <Part BlockName="B2"/>
      <CuttingParams>
        <CutBlock BlockName="B2.1" Box="0 0 800 300"/>
        <CutBlock BlockName="B2.2" Box="0 300 800 600"/>
      </CuttingParams>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Component" Usage="Input">
    <Resource>
      <Component Dimensions="1200 600 0"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Component" Usage="Output">
    <Resource>
      <Part BlockName="B1.1"/>
      <Component Dimensions="400 300 0"/>
    </Resource>
    <Resource>
      <Part BlockName="B1.2"/>
      <Component Dimensions="400 300 0"/>
    </Resource>
    <Resource>
      <Part BlockName="B2.1"/>
      <Component Dimensions="800 300 0"/>
    </Resource>
    <Resource>
      <Part BlockName="B2.2"/>
      <Component Dimensions="800 300 0"/>
    </Resource>
  </ResourceSet>
</XJDF>

```

6.25.1 CutBlock

CutBlock specifies exactly one cut block on a sheet. The **CutBlock** SHALL be defined in the coordinate system of the input **Component**.

Table 6.53: CutBlock Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>BinderySignatureIDs</i> ?	NMTOKENS	If specified, @ <i>BinderySignatureIDs</i> SHALL list the <i>BinderySignature</i> / <i>@BinderySignatureID</i> of all <i>BinderySignatures</i> that comprise this CutBlock .
<i>BlockName</i>	NMTOKEN	Name of the block. The output Component of the Cutting process SHALL be partitioned by <i>Part</i> / <i>@BlockName</i> . The values of <i>Part</i> / <i>@BlockName</i> SHALL match the value of this @ <i>BlockName</i> .
<i>Box</i> ?	rectangle	Defines the position and size of the block relative to the parent Component coordinate system.

Table 6.53: CutBlock Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>CutWidth</i> ?	float	Width in points of the u-shaped knife, saw blade, etc.

6.26 DeliveryParams

DeliveryParams provides information needed by a **Delivery** process. A **Delivery** process is the sending or receiving of one or more products to one or more delivery destinations. Delivery is also used to specify the scheduled transfer of digital assets.

Resource Properties

Input of Processes: **Delivery**

Table 6.54: DeliveryParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>BuyerAccount</i> ?	string	Account ID of the buyer with the delivery service.
<i>Earliest</i> ?	dateTime	@ <i>Earliest</i> SHALL specify the earliest date and time after which the delivery is intended to be made.
<i>EarliestDuration</i> ?	duration	@ <i>EarliestDuration</i> SHALL specify the earliest duration by which the delivery SHALL be made relative to the date and time that the order is ready for production.
<i>Method</i> ?	NMTOKEN	Specifies a delivery method. Values include those from: ▶ Delivery Methods.
<i>Ownership</i> ?	enumeration	Point of transfer of ownership. Allowed values are: Destination – Ownership is transferred upon receipt at destination. Origin – Ownership of goods is transferred upon leaving point of origin.
<i>Required</i> ?	dateTime	@ <i>Required</i> SHALL specify the date and time by which the delivery is intended to be made.
<i>RequiredDuration</i> ?	duration	@ <i>RequiredDuration</i> SHALL specify the time duration by which the delivery SHALL be made relative to the date and time that the order is ready for production.
<i>TrackingID</i> ?	string	The carrier's identifier for the delivery. The format of @ <i>TrackingID</i> is determined by the carrier chosen to ship the product(s).
<i>Transfer</i> ?	enumeration	Describes the direction and responsibility of the transfer. Allowed values are: BuyerToPrinterDeliver – 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. BuyerToPrinterPickup – 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. PrinterToBuyerDeliver – The DeliveryParams describes an output of the job. In this case, the printer delivers the merchandise to the buyer. PrinterToBuyerPickup – The DeliveryParams describes an output of the job. In this case, the buyer picks up the merchandise.
<i>Dropltem</i> *	element	A delivery MAY consist of multiple products. Each Dropltem describes an individual Resource or Product that is part of this delivery.
FileSpec (DeliveryContents) ?	element	A FileSpec element that references a document that SHALL be printed and packaged together with the delivered items.
FileSpec (MailingList) ?	element	A FileSpec element pointing to a mailing list. The format of the referenced mailing list is implementation dependent.
FileSpec (RemoteURL) ?	element	A FileSpec element that specifies the remote location of a digital delivery.

6.26.1 Dropltem

Table 6.55: Dropltem Element

NAME	DATA TYPE	DESCRIPTION
<i>Amount</i>	integer	@Amount SHALL be present and specify the number of products or resources that comprise this Dropltem .
<i>ItemRef</i>	IDREF	@ItemRef SHALL reference the Resource , ResourceSet , ProofItem or ProductList/Product that represents the individual items to be delivered. If the Dropltem references a final product, @ItemRef SHALL reference a ProductList/Product .
<i>TotalDimensions</i> ?	shape	Total dimensions in points of all individual items, including packaging.
<i>TotalVolume</i> ?	float	Total volume in liters of all individual items, including packaging.
<i>TotalWeight</i> ?	float	Total weight of all individual items, including packaging.

6.27 DevelopingParams

DevelopingParams specifies information about the chemical and physical properties of the developing and fixing process for film and plates. Includes details of preheating, post-baking and post-exposure.

- Preheating is necessary for negative working plates. It hardens the exposed areas of the plate to make it durable for the subsequent developing process. The stability and uniformity of the preheat temperature influence the evenness of tints and the run length of the plate on press.
- Post-baking is an optional process of heating that is applied to most polymer plates to enhance the run length of the plate. A factor of 5 to 10 can be gained compared to plates that are not post-baked.
- Post-exposure 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 post-exposed.

Resource Properties

Input of Processes: **ImageSetting**

Table 6.56: DevelopingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>PostBakeTemp</i> ?	float	Temperature of the post-baking process in °C. @PostBakeTemp SHALL NOT be specified if @PostExposeTime is present.
<i>PostBakeTime</i> ?	duration	Duration of the post-baking process. @PostBakeTime SHALL NOT be specified if @PostExposeTime is present.
<i>PostExposeTime</i> ?	duration	Duration of the post-exposing process. @PostExposeTime SHALL NOT be specified if @PostBakeTime is present.
<i>PreHeatTemp</i> ?	float	Temperature of the preheating process in °C.
<i>PreHeatTime</i> ?	duration	Duration of the preheating process.

6.28 Device

Device describes the physical properties of the main device that executes an **XJDF** process. See ▶ Chapter 5 Processes. Examples are a press or a finishing machine. See **Tool** for a description of auxiliary devices such as fork lifts.

Resource Properties

Resource referenced by: **DieLayout**, **DieLayoutProductionParams/ConvertingConfig**, **InkZoneCalculationParams**, **Layout**, **ResponseKnownDevices**, **SignalKnownDevices**

Input of Processes: **Any Process**

Table 6.57: Device Resource (Sheet 1 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>CostCenterID</i> ?	NMTOKEN	MIS cost center ID.

Table 6.57: Device Resource (Sheet 2 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>DescriptiveName</i> ?	string	Human readable description of the device.
<i>DeviceClass</i> ?	NMTOKENS	Indicates the class of device. Multiple NMTOKENS SHALL be used to describe integrated devices with multiple classes. Values include those from: ▶ Device Classes.
<i>DeviceID</i>	NMTOKEN	Identifier of the device. <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 <i>XJMF/@DeviceID</i> and for <i>@DeviceID</i> 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>ICSVersions</i> ?	NMTOKENS	CIP4 Interoperability Conformance Specification (ICS) Versions that this Device complies with. The value of <i>@ICSVersions</i> SHALL conform to the value format described in ▶ Section 3.1.1 ICS Versions Value.
<i>JDFVersions</i> ?	enumerations	Whitespace separated list of supported JDF and XJDF versions that this device supports. Allowed values are: 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 1.6 - JDF 1.6 2.0 - XJDF 2.0, this specification.
<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>Packaging</i> ?	enumerations	List of packaging methods supported for job submission and job return with <i>SubmitQueueEntry</i> , <i>ResubmitQueueEntry</i> and <i>ReturnQueueEntry</i> . Allowed values are: XML – Unpackaged XML is supported. Zip – Zip packaging of XJMF , XJDF and digital resources is supported. See ▶ Section 9.7 XJDF Packaging for details.
<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>Revision</i> ?	string	Hardware or software version of the Device . Note: <i>@SerialNumber</i> is independent of upgrades whereas <i>@Revision</i> SHOULD be modified when hardware or software is changed. Note: <i>Header/@AgentVersion</i> is the version of the XJDF interpreter whereas <i>@Revision</i> applies to the hardware or software of the underlying ▶ Machine.
<i>SerialNumber</i> ?	string	Serial number of the device.
<i>URLSchemes</i> ?	NMTOKENS	List of schemes supported for retrieving XJDF files. If not specified, the controller does not support retrieving XJDF files from remote URLs. Values include: 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).

Table 6.57: Device Resource (Sheet 3 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>XJMFURL</i> ?	URL	URL of the device port that will accept XJMF messages. A controller that manages a device MAY specify its own <i>@XJMFURL</i> when responding to <i>KnownDevices</i> messages.
<i>IconList</i> ?	element	List of locations of icons that can be used to represent the <i>Device</i> .
<i>IdentificationField</i> *	element	<i>IdentificationField</i> associates bar codes or labels with this <i>Device</i> .
<i>Module</i> *	element	Individual modules that are represented by this <i>Device</i> .

6.28.1 Icon

An *Icon* represents a device in the user interface.

Table 6.58: Icon Element

NAME	DATA TYPE	DESCRIPTION
<i>BitDepth</i>	integer	Bit depth of one color.
<i>IconUsage</i> ?	enumerations	The <i>DeviceInfo</i> / <i>@Status</i> of the device that this <i>Icon</i> represents. If not specified, the icon is independent of the status of the device. Allowed values are from: ▶ <i>DeviceStatus</i> .
<i>Size</i>	XYPair	Height and width of the icon in pixels.
<i>FileSpec</i> ?	element	Reference to details of the icon data.

6.28.2 IconList

The *IconList* is a list of individual icon descriptions.

Table 6.59: IconList Element

NAME	DATA TYPE	DESCRIPTION
<i>Icon</i> +	element	Individual icon description.

6.28.3 Module

A *Module* represents a physical machine or part of a *Device*.

Table 6.60: Module Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>DescriptiveName</i> ?	string	Human readable description of the module.
<i>Manufacturer</i> ?	string	Manufacturer name.
<i>ManufacturerURL</i> ?	URL	Web site for manufacturer.
<i>ModuleID</i>	NMTOKEN	Identifier of the module. This is a unique identifier within the workflow. <i>@ModuleID</i> SHALL be the same over time for a specific <i>Module</i> instance (i.e., SHALL survive reboots). If multiple logical devices share a physical <i>Module</i> , <i>@ModuleID</i> SHALL be identical. <i>@ModuleID</i> SHOULD be used to specify machines that comprise a <i>Device</i> .
<i>ModuleType</i> ?	NMTOKENS	<i>@ModuleType</i> provides a classification of the module. Modules with multiple functions SHOULD provide all appropriate values. If known, <i>@ModuleType</i> SHOULD be provided. Values include those from: ▶ <i>Module Types</i> .
<i>Revision</i> ?	string	Hardware or software version of the <i>Module</i> . See <i>Device</i> / <i>@Revision</i> .

Table 6.60: Module Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>SerialNumber</i> ?	string	Serial number of the <i>Module</i> .
<i>IdentificationField</i> *	element	<i>IdentificationField</i> associates bar codes or labels with this <i>Module</i> .

6.29 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. 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 6.61: DieLayout Resource

NAME	DATA TYPE	DESCRIPTION
<i>CutBox</i> ?	rectangle	A rectangle describing the bounding box of all cut lines in the <i>DieLayout</i> . This is sometimes referred to as the knife to knife dimensions of the <i>DieLayout</i> . If the position on the <i>Media</i> is not known, the lower left SHOULD be set to 0 0.
<i>CutLines</i> ?	NMTOKENS	Selects the die line separation identifiers from the file referenced by <i>FileSpec</i> . Additional details of the usage of the separations MAY be specified in the respective <i>ResourceSet</i> [@Name="Color"].
<i>DieSide</i> ?	enumeration	Determines the die side for which the <i>DieLayout</i> is made. Allowed values are: <i>Down</i> – The <i>DieLayout</i> is made with the knives pointing downwards. <i>Up</i> – The <i>DieLayout</i> is made with the knives pointing upwards.
<i>MediaSide</i> ?	enumeration	Determines the printing side for which the <i>DieLayout</i> is made. "Front" corresponds to the outside of a box, "Back" corresponds to the inside of a box. Allowed value is from: ▶ Side.
<i>Rotated</i> ?	boolean	Indicates if some of the structural designs are oriented cross grain/flute in the layout.
<i>Waste</i> ?	float	The percent of the material that is wasted. Inner waste, i.e. cut out windows, are not included in the waste.
<i>Device</i> *	element	The devices for which this <i>DieLayout</i> was made (printing press and die cutter). Typically only the type of device would be used (e.g., the model of the die cutter).
<i>FileSpec</i> ?	element	Reference to an external URL that represents the die. This is typically a CAD design file.
<i>Media</i> ?	element	<i>Media</i> for which this <i>DieLayout</i> was intended. The <i>Media</i> description defines important design parameters as the type of <i>Media</i> , dimensions, grain direction or flute direction.
<i>RuleLength</i> *	element	Elements describing the length of die rules for the different types of rules. Each <i>RuleLength</i> element describes the accumulated length of all rules of a certain type.
<i>Station</i> *	element	Description of the stations in a <i>DieLayout</i> . One <i>Station</i> produces one shape.

6.29.1 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 6.62: RuleLength Element

NAME	DATA TYPE	DESCRIPTION
<i>DDESCutType</i>	integer	Type of rule. Values for <i>@DDESCutType</i> SHALL be between "0" and "999". These values correspond to the line type as defined in ▶ [DDES3].
<i>Length</i>	float	Accumulated length of the rules of this type in the <i>DieLayout</i> , in points.

6.29.2 Station

Description of an individual 1-up station in a *DieLayout*. One station produces one shape.

Table 6.63: Station Element

NAME	DATA TYPE	DESCRIPTION
<i>BinderySignatureIDs</i> ?	NMTOKENS	If specified, <i>@BinderySignatureIDs</i> SHALL list the <i>BinderySignature/BinderySignatureID</i> of all <i>BinderySignatures</i> that are processed by this <i>Station</i> .
<i>ShapeDefRef</i> ?	IDREF	If present, <i>@ShapeDefRef</i> SHALL reference a <i>ShapeDef</i> that defines the shape of the <i>Station</i> in the <i>DieLayout</i> .
<i>StationName</i>	NMTOKEN	The name of the 1-up design in the <i>DieLayout</i> . Multiple stations with the same <i>@StationName</i> MAY be specified.

6.30 DieLayoutProductionParams

Parameters for the die layout.

Resource Properties

Input of Processes: **DieLayoutProduction**

Table 6.64: DieLayoutProductionParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>Estimate</i> ?	boolean	Determines if the process SHALL run in estimate mode or not. When in estimate mode multiple solutions SHOULD be generated.
<i>Position</i> ?	enumeration	The position of the <i>DieLayout</i> on the sheet. Allowed value is from: ▶ Anchor.
<i>ConvertingConfig</i> +	element	A <i>ConvertingConfig</i> element describes a range of sheet sizes that can be taken into account to create a new <i>DieLayout</i> . Typically a <i>ConvertingConfig</i> will correspond to a single combination of printing press and further finishing equipment such as die cutters.
<i>RepeatDesc</i> +	element	Step and repeat parameters for a set of <i>ShapeDef</i> .

6.30.1 RepeatDesc

The *RepeatDesc* element describes the layout specs for a *ShapeDef*.

Table 6.65: RepeatDesc Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>AllowedRotate</i> ?	enumeration	Allowed methods to rotate structural designs 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.

Table 6.65: RepeatDesc Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>GutterX</i> ?	float	Gutter between columns (see also <i>@GutterX2</i>).
<i>GutterX2</i> ?	float	Secondary gutter between columns. The gutter between columns (2n+0) and (2n+1) is <i>@GutterX</i> and between columns (2n+1) and (2n+2) is <i>@GutterX2</i> . When <i>@GutterX2</i> is not specified <i>@GutterX2=@GutterX</i> . See ▶ Figure 6-20: RepeatDesc using Secondary Gutters.
<i>GutterY</i> ?	float	Gutter between rows (see also <i>@GutterY2</i>).
<i>GutterY2</i> ?	float	Secondary gutter between rows. The gutter between rows (2n+0) and (2n+1) is <i>@GutterY</i> and between rows (2n+1) and (2n+2) is <i>@GutterY2</i> . When <i>@GutterY2</i> is not specified <i>@GutterY2=@GutterY</i> . See ▶ Figure 6-20: RepeatDesc using Secondary Gutters.
<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 6-14: Basic Shape for RepeatDesc/ <i>@LayoutStyle</i> examples 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 SHALL be present when a layout is being made for more than one <i>ShapeDef</i> .
<i>ShapeDefRef</i>	IDREF	Reference to a <i>ShapeDef</i> describing the 1-up structural design that SHALL to be stepped and repeated on the <i>DieLayout</i> .
<i>UseBleed</i> ?	boolean	If true, the print bleed defined in the structural design SHALL be used to calculate the layout. If false, the outer cut SHALL be used.

The following figure shows the basic shape for subsequent figures that relate to *RepeatDesc*.

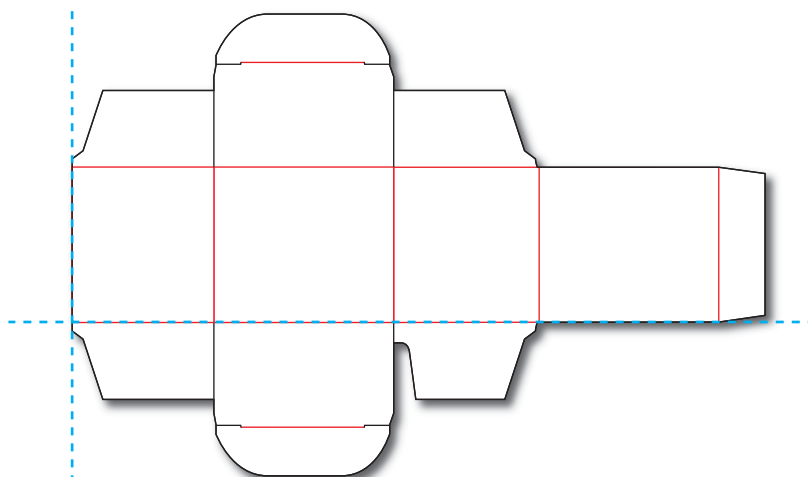
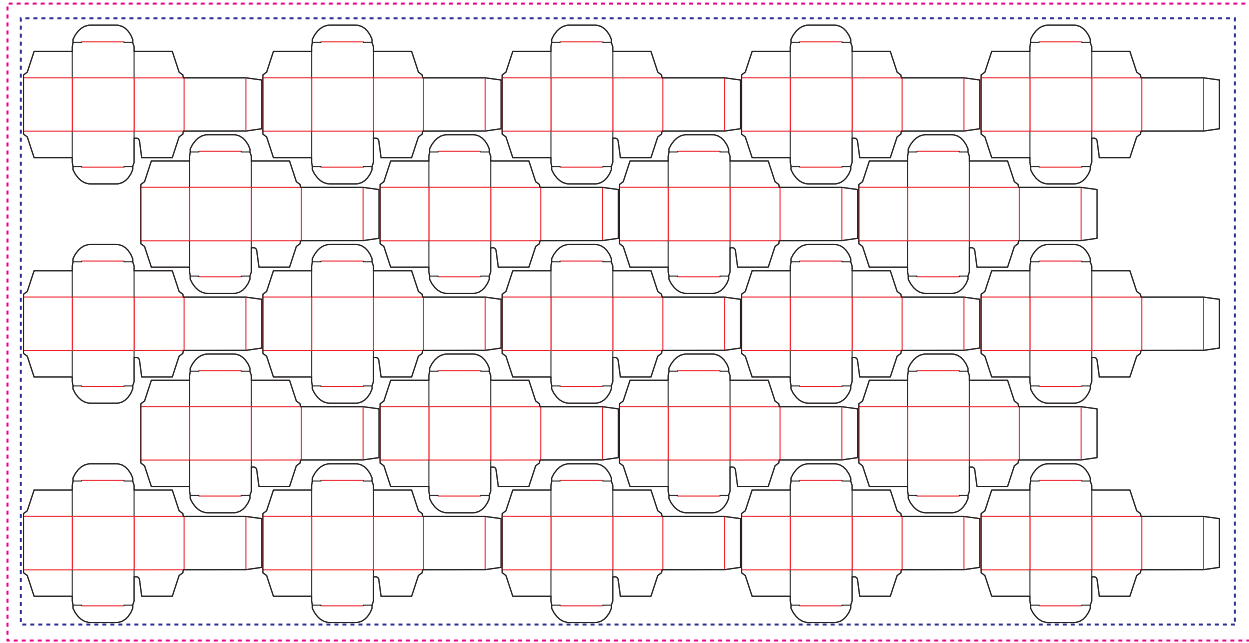
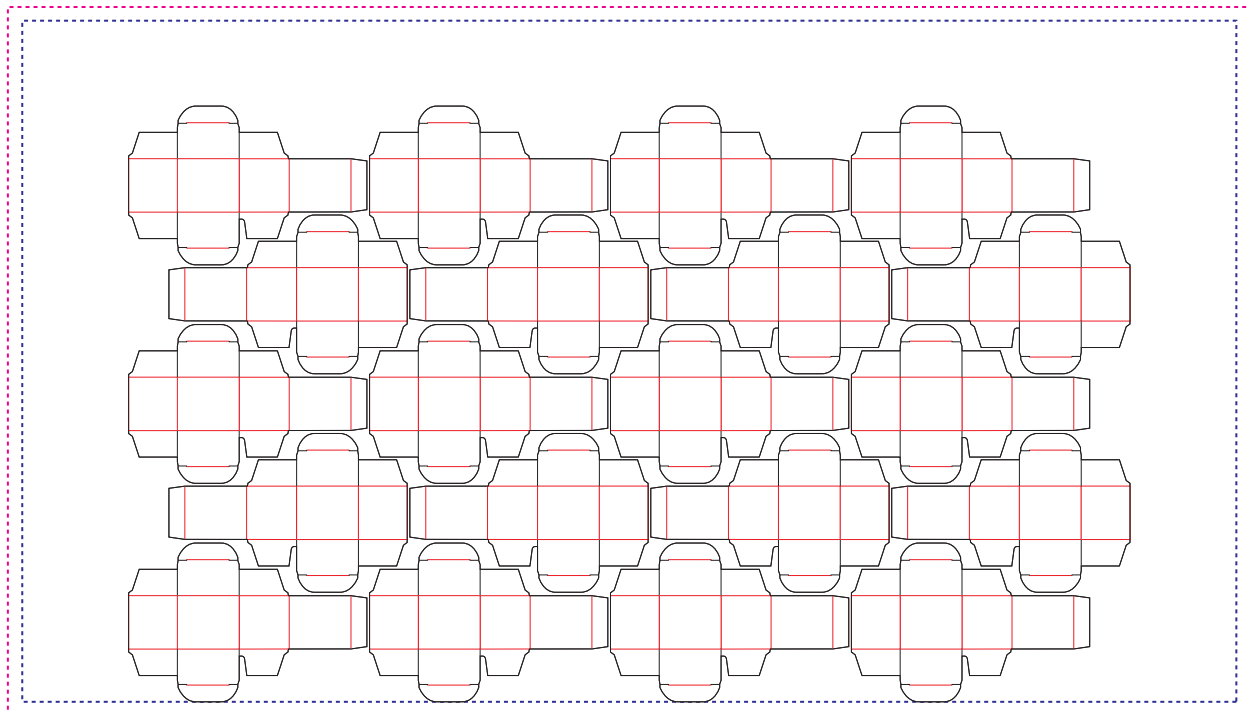
Figure 6-14: Basic Shape for RepeatDesc/*@LayoutStyle* examples

Figure 6-15: RepeatDesc/@LayoutStyle = "StraightNest"



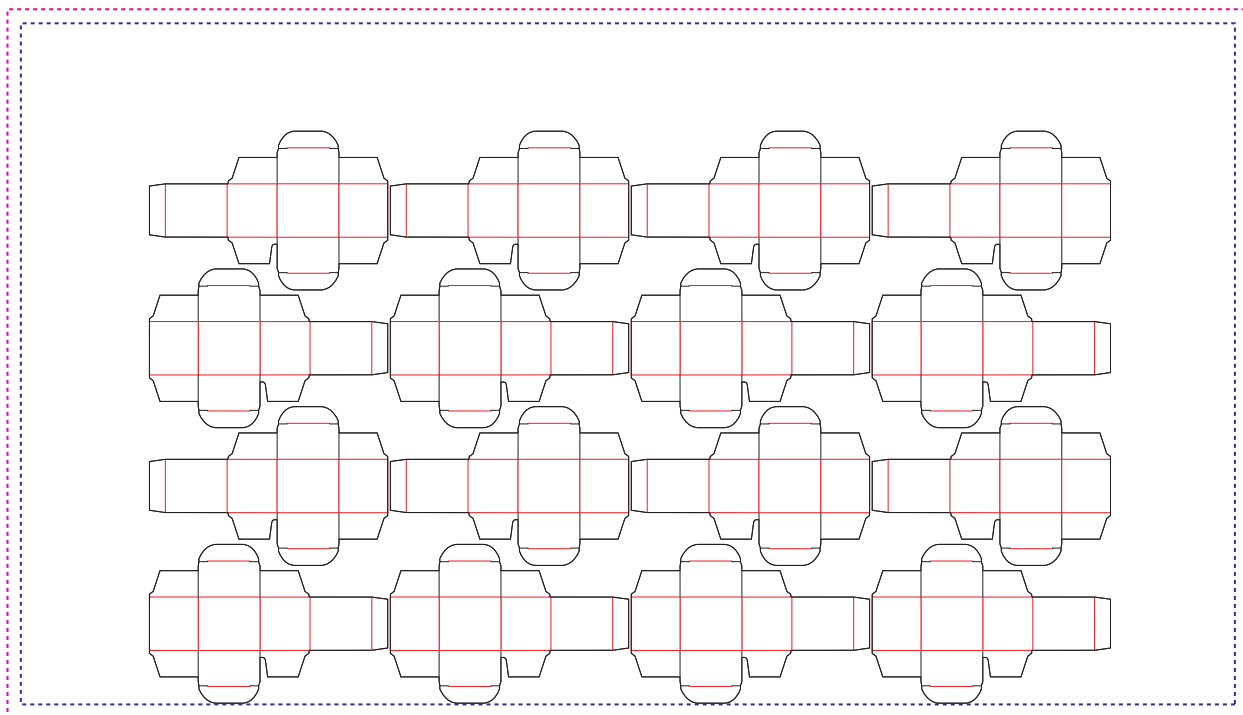
In the following figure, 1-ups on even rows are rotated 180 degrees. Even rows are shifted horizontally and vertically to obtain optimal nesting.

Figure 6-16: RepeatDesc/@LayoutStyle = "Reverse2ndRow"



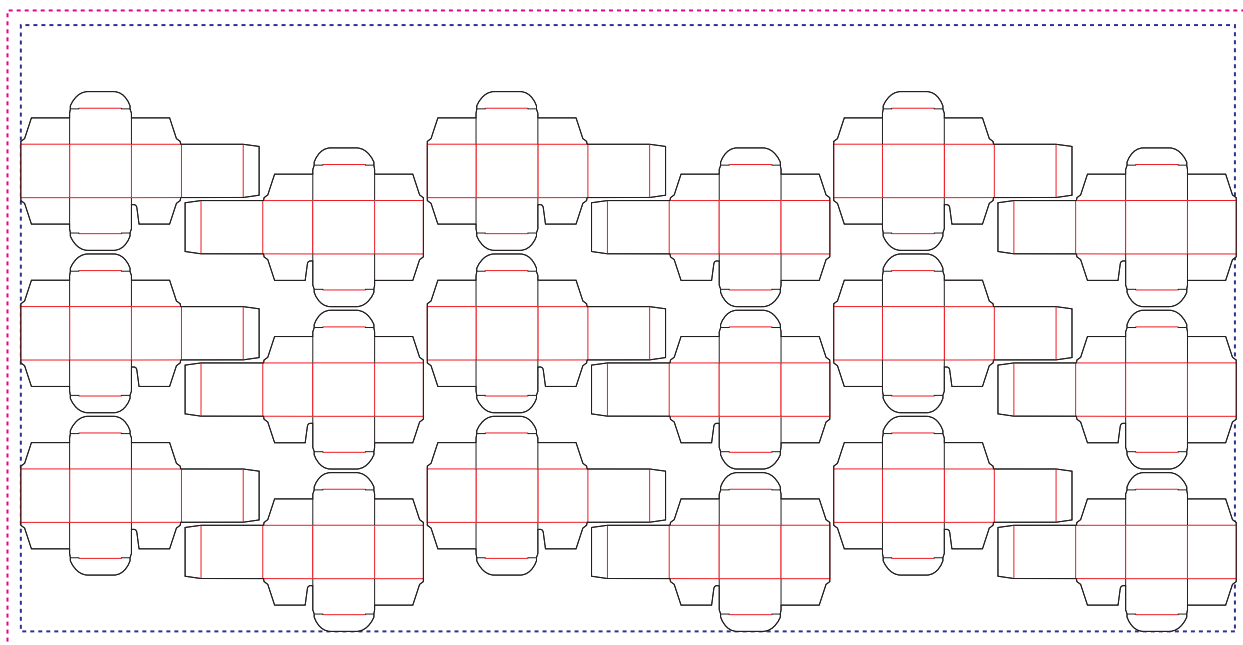
In the following figure, 1-ups on even rows are rotated 180 degrees. Even rows are shifted vertically to obtain optimal nesting. The even rows are not shifted horizontally (left and right edges are aligned between rows).

Figure 6-17: 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.

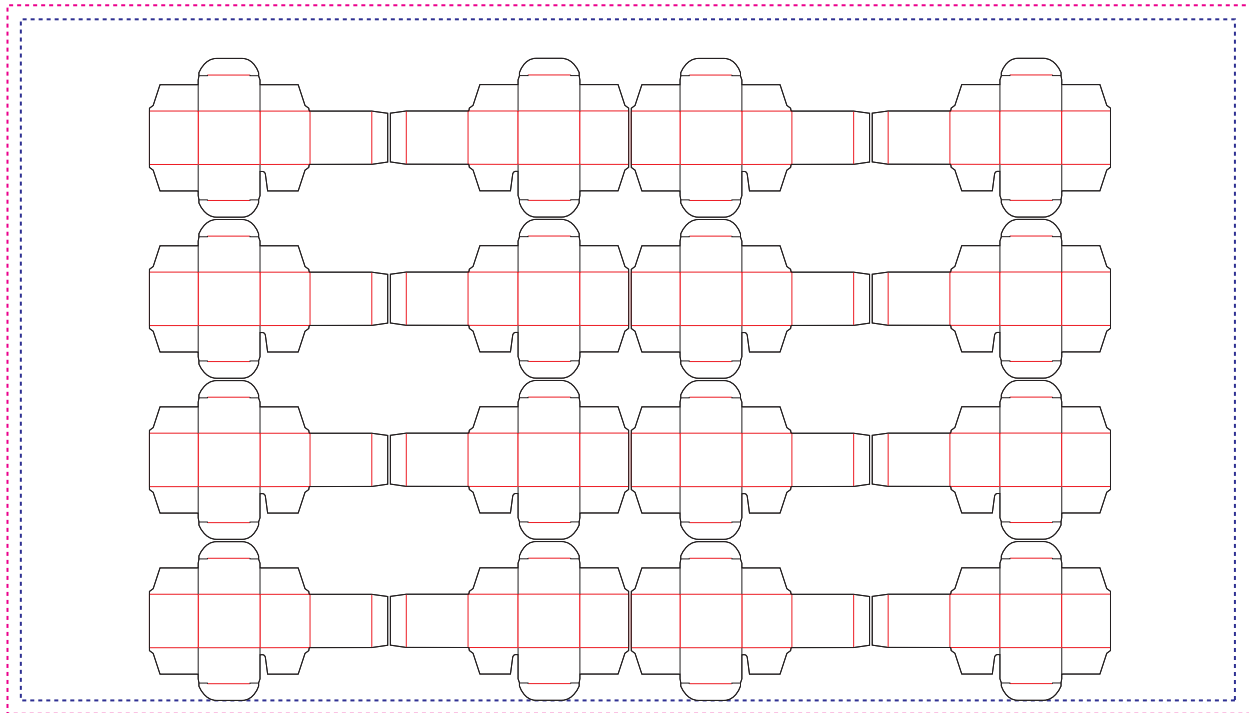
Figure 6-18: RepeatDesc/@LayoutStyle = "Reverse2ndColumn"



RESOURCES

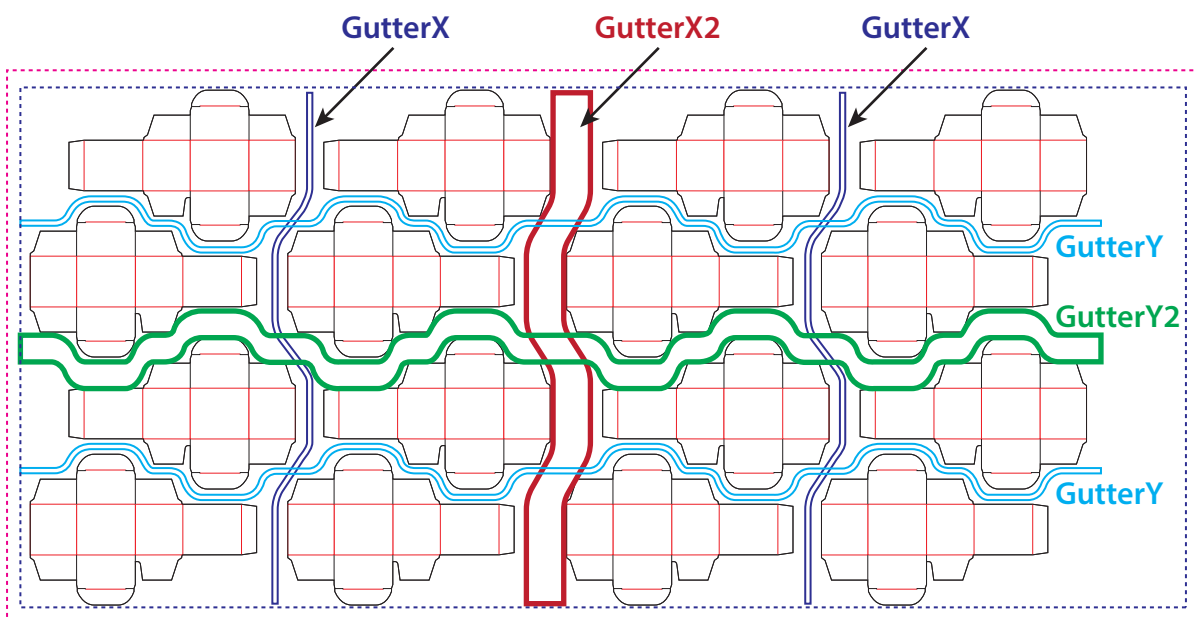
In the following figure, 1-ups on even columns are rotated 180 degrees. Even columns are shifted horizontally to obtain optimal nesting. No vertical shifting of even columns is done (top and bottom edges are aligned between columns).

Figure 6-19: RepeatDesc/@LayoutStyle = "Reverse2ndColumnAligned"



In the following figure, @LayoutStyle="Reverse2ndRow", @GutterX="36", @GutterX2="70", @GutterY="20", @GutterY2="40".

Figure 6-20: RepeatDesc using Secondary Gutters



6.31 DigitalPrintingParams

DigitalPrintingParams contains details of the **DigitalPrinting** process.

6.31.1 Coordinate systems in DigitalPrinting

► Section 2.6 Coordinate Systems in XJDF defines the coordinate system for **ConventionalPrinting** and **DigitalPrinting**.
Note: The paper feed direction of the idealized process is towards the X-axis, which corresponds to bottom edge first.

Properties

Intent Pairing: **VariableIntent**

Input of Processes: **DigitalPrinting**

Table 6.66: DigitalPrintingParams Resource

NAME	DATA TYPE	DESCRIPTION
Collate ?	enumeration	Determines the sequencing of the printed sheets when multiple copies are requested as output. Allowed values are: None – Do not collate. All copies of a sheet SHALL be printed prior to printing the next sheet. Sheet – Collate. All sheets in one document SHALL be printed in sequence prior to printing the next document.
ManualFeed ?	boolean	Indicates whether the media will be fed manually.
PageDelivery ?	enumeration	Indicates how pages SHALL be delivered to the output bin or finisher. Note: These values refer to the orientation of the entire stack being output from the press, not individual sheets. For example, " SameOrderFaceDown " 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. Allowed values are: FanFold – The output is alternating face-up, face down. SameOrderFaceUp – Order as defined by the RunList , with the front sides of the media up and the first sheet on top. SameOrderFaceDown – Order as defined by the RunList , with the front sides of the media down and the first sheet on the bottom. ReverseOrderFaceUp – Sheet order reversed compared to " SameOrderFaceUp ", with the front sides of the media up and the last sheet on top. ReverseOrderFaceDown – Sheet order reversed compared to " SameOrderFaceDown ", with the front sides of the media down and the last sheet on the bottom.
SheetLay ?	enumeration	Lay of input media. Reference edge where paper is placed in feeder. Allowed value is from: ▶ SheetLay .
Sides ?	enumeration	Indicates whether the ByteMap SHALL be imaged on one or both sides of the media. If the RunList input to DigitalPrinting is partitioned by @Side then the input RunList provides a binding of front and back surfaces to sheets. 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 were not present. Allowed values are: OneSidedBack OneSidedFront TwoSided Note: The orientation of the front pages relative to back pages SHOULD be completely defined in the explicit or implied imposition Layout .

6.32 EmbossingParams

[EmbossingParams](#) contains attributes and elements used in executing the **Embossing** process. **Embossing** can also be used to model a foil stamping process.

Resource Properties

Intent Pairing: [EmbossingIntent](#)Input of Processes: **Embossing**

Table 6.67: EmbossingParams Resource

NAME	DATA TYPE	DESCRIPTION
ModuleID ?	NMTOKEN	Identifier of the embossing module in a multi-function device such as a printing press. See Module for details.
Emboss +	element	One Emboss element is specified for each impression.

6.32.1 Emboss

Table 6.68: Emboss Element

NAME	DATA TYPE	DESCRIPTION
<i>Direction</i>	enumeration	The direction of the image. Allowed value is from: ▶ EmbossDirection.
<i>EdgeAngle</i> ?	float	The angle of a beveled edge in degrees. Typical values are angles of: 30, 40, 45, 50 or 60 degrees. If @EdgeAngle is specified, @EdgeShape="Beveled" SHALL be specified.
<i>EdgeShape</i> ?	enumeration	The transition between the embossed surface and the surrounding media can be rounded or beveled (angled). Allowed values are: Beveled Rounded
<i>EmbossingType</i>	enumeration	Specifies the type of embossing required. Allowed value is from: ▶ EmbossType.
<i>Face</i> ?	enumeration	Position of the embossing on the product. Allowed value is from: ▶ Face.
<i>Height</i> ?	float	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.
<i>Position</i> ?	XYPair	Position of the lower left corner of the bounding box of the embossed image in the coordinate system of the surface of the Component that is selected by @Face.
<i>ToolRef</i> ?	IDREF	@ToolRef SHALL reference the Tool that is used to make the embossing described by this Emboss .
<i>IdentificationField</i> ?	element	If @EmbossingType="Braille", IdentificationField SHALL describe the content of the Braille element.

6.33 EndSheetGluingParams

EndSheetGluingParams describes the attributes and elements used in executing the **EndSheetGluing** process.

EndSheetGluingParams SHOULD be partitioned by **Part**/@Side.

Resource Properties

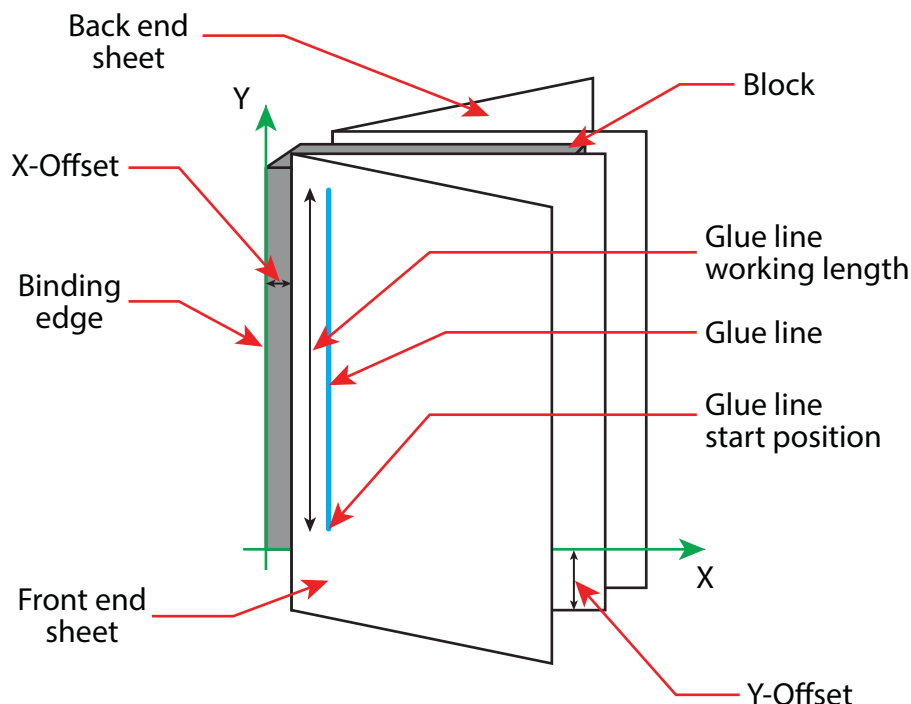
Intent Pairing: **BindingIntent**

Input of Processes: **EndSheetGluing**

Table 6.69: EndSheetGluingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>Glue</i>	element	Description of the glue that is used to attach the end sheet to the cover.

Figure 6-21: Parameters and coordinate system used for EndSheetGluing



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 side to the face side opposite the binding side.

6.34 ExposedMedia

ExposedMedia represents processed **Media** such as film or plate. The **@ExternalID** attribute in the parent **Resource** element SHALL be unique within the workflow.

Resource Properties

Input of Processes: **Bending, ConventionalPrinting, Varnishing**

Output of Processes: **Bending, ImageSetting**

Table 6.70: ExposedMedia Resource

NAME	DATA TYPE	DESCRIPTION
<i>MediaRef</i>	IDREF	@MediaRef SHALL reference a Media that specifies details of the original media such as size.
<i>PlateType</i> ?	enumeration	Specifies whether a plate is exposed or a dummy plate. Allowed values are: Dummy – Specifies a dummy plate that has not been imaged. Usually, dummy plates are only needed on newspaper/web presses or for Varnishing . Exposed – The plate has been imaged.
<i>Polarity</i> ?	enumeration	@Polarity specifies the polarity of the image on the ExposedMedia . Allowed value is from: ▶ Polarity.
<i>PunchType</i> ?	NMTOKEN	Name of the registration punch scheme. See Bending . If not specified, no holes have been punched. Values include: Bacher Stoesser
<i>Resolution</i> ?	XYPair	Resolution of the output.
<i>IdentificationField</i> *	element	IdentificationField associates bar codes or labels with this ExposedMedia .

6.35 FeedingParams

The parameters for any **XJDF** feeder processing device.

Resource Properties

Input of Processes: **Feeding**

Table 6.71: FeedingParams Resource

NAME	DATA TYPE	DESCRIPTION
CollatingItem *	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.
Feeder *	element	Defines the specifics of an individual Feeder . If a Component from the input resource list is not referenced from a Feeder in this list, a system defined Feeder will be used.

6.35.1 CollatingItem

Table 6.72: CollatingItem Element

NAME	DATA TYPE	DESCRIPTION
Amount ?	integer	Determines how many consecutive items shall be consumed.
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 ?	enumeration	Named orientation of the CollatingItem relative to the input coordinate system. For details see ▶ Table 2.1 Matrices and Orientation values for describing the orientation of a Component. At most one of @Orientation or @Transformation SHALL be specified. If neither is specified, no transformation is applied. The transformation specified here is applied in addition to any orientation/transformation specified in the respective Resource . Allowed value is from: ▶ Orientation.
Transformation ?	matrix	Transformation of the CollatingItem relative to the input coordinate system. For details see ▶ Table 2.1 Matrices and Orientation values for describing the orientation of a Component. At most one of @Orientation or @Transformation SHALL be specified. If neither is specified, no transformation is applied. The transformation specified here is applied in addition to any orientation/transformation specified in the respective Resource .
TransformationContext ?	enumeration	This parameter specifies the object that SHALL be manipulated in orientation/transformation, and it is important to determine the sequence of stack items after flipping. Allowed values are: CollateItem – Apply to a CollatingItem as a whole. Component – Apply to each single element of a CollatingItem individually. StackItem – Apply individually to the smallest element on the stack that can be manipulated individually (e.g., to a single sheet in the case of a stack of sheets). Note: If @Amount="1" , 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, and a flipping mechanism for the **Winding** process.

6.35.2 Feeder

Table 6.73: Feeder Element

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	References the <i>Component</i> that SHALL be loaded into this <i>Feeder</i> . This <i>Component</i> SHALL be an input of the <i>Feeding</i> process.
<i>FeederSynchronization</i> ?	enumeration	Specifies the synchronization of multiple <i>Feeder</i> elements with identical <i>Component</i> elements. Allowed values are: <i>Alternate</i> – The feeders specified in <i>@Position</i> SHALL alternate. <i>Backup</i> – This feeder is the backup feeder for the <i>Component</i> in case of a mis-feed or malfunction. The priority of backup feeders SHALL be 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 <i>Component</i> .
<i>FeederType</i> ?	NMTOKEN	Specifies the feeder type. Values include: <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 <i>Component</i> . <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	Specifies the feeder loading. Values include: <i>Bundle</i> – Stream feeder, using the output of the <i>Bundling</i> process. <i>FanFold</i> – Automatic loading of fan fold media. <i>Manual</i> – Manual loading of stacks. <i>Online</i> – Loaded by a gripper or conveyor. The "Online" 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 <i>Winding</i> process.
<i>Opening</i> ?	enumeration	Specifies the opening of signatures. Allowed values are: <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 is required.
<i>Position</i> ?	integer	<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 <i>Feeder</i> SHALL 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., "-1" = last position, "-2" = next to last position, etc.).
<i>FeederQualityParams</i> ?	element	Definition of the setup and policy for feeding quality.

6.35.3 FeederQualityParams

The **FeederQualityParams** element defines the setup and policy for feeding quality control. It is specified individually for each **Feeder**.

Table 6.74: FeederQualityParams Element

NAME	DATA TYPE	DESCRIPTION
<i>BadFeedQuality</i> ?	enumeration	Defines the operation of the bad feed quality control. Allowed value is from: ▶ FeedQuality.
<i>BadFeeds</i> ?	integer	Number of consecutive bad feeds until the device SHALL stop.
<i>DoubleFeedQuality</i> ?	enumeration	Defines the operation of the double feed quality control. Allowed value is from: ▶ FeedQuality.
<i>DoubleFeeds</i> ?	integer	Number of consecutive double feeds until the device SHALL stop.
<i>IncorrectComponent Quality</i> ?	enumeration	Defines the operation of the incorrect components quality control. Allowed value is from: ▶ FeedQuality.
<i>IncorrectComponents</i> ?	integer	Number of consecutive incorrect components until the device SHALL stop.

6.36 FoldingParams

FoldingParams describes the folding parameters, including the sequence of folding steps. After each folding step of a folding procedure, the origin of the coordinate system SHALL be moved to the lower left corner of the intermediate folding product.

The order of the **Crease**, **Cut**, **Fold** and **Perforate** elements in ▶ Table 6.75 FoldingParams Resource SHALL specify the order of operations in the machine. Therefore any of the four element types MAY be specified in any order.

Resource Properties

Intent Pairing: **FoldingIntent**

Input of Processes: **Folding**

Table 6.75: FoldingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>FoldCatalog</i> ?	NMTOKEN	Describes the type of fold. Values include those from: ▶ Fold Catalog.
<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 value is from: ▶ SheetLay.
FileSpec (CIP3) ?	element	Reference to a CIP3 file that contains folding instructions in the ▶ [CIP3 - PPF] format. If FileSpec (CIP3) is specified, Crease , Cut , Fold and Perforate SHALL NOT be present.
Crease *	element	Defines one or more Crease lines.
Cut *	element	Cut elements describe an individual cut.
Fold *	element	Describes the folding operations in the sequence in which they are to be carried out. If both @ <i>FoldCatalog</i> and Fold elements are specified, the Fold 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 "F4-1".
Perforate *	element	Defines one or more Perforate lines.

6.37 FontPolicy

FontPolicy defines the policies that devices SHALL 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 that 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 process SHALL fail if a referenced font is not provided.

Resource Properties

Input of Processes: **Interpreting, Trapping**

Table 6.76: FontPolicy Resource

NAME	DATA TYPE	DESCRIPTION
<code>PreferredFont</code>	NMTOKEN	The name of a font that SHALL be used as the default font for this job.
<code>UseDefaultFont</code>	boolean	If "true", the device SHALL resort to a default font if a font cannot be found. Note: This is the normal behavior of the PostScript interpreter, which defaults to Courier when a font cannot be found.
<code>UseFontEmulation</code>	boolean	If "true", the device SHALL emulate a requested font if a font cannot be found.

6.38 GluingParams

GluingParams define the parameters for applying a generic line of glue to a component.

Resource Properties

Intent Pairing: **BindingIntent**

Input of Processes: **Gluing**

Table 6.77: GluingParams Resource

NAME	DATA TYPE	DESCRIPTION
<code>GluingProductionID</code> ?	NMTOKEN	Defines a gluing scheme for production.
<code>Glue</code> *	element	Definition of one or more Glue line applications.

6.39 HeadBandApplicationParams

HeadBandApplicationParams specifies how to apply head bands in hardcover book production.

Resource Properties

Input of Processes: **HeadBandApplication**

Table 6.78: HeadBandApplicationParams Resource

NAME	DATA TYPE	DESCRIPTION
<code>Length</code> ?	float	Length of the carrier material of the head band along the binding edge. If not specified the head band has the length of the book block. If both top and bottom head bands are specified in a partitioned ResourceSet (using Part/@Option), then: <ul style="list-style-type: none"> • If <code>@Length</code> is not specified, then both head bands SHALL be on one carrier that has the length of the book block. • If <code>@Length</code> is provided in only one of the partitions, then both head bands are on one carrier of the given length. • If <code>@Length</code> is provided in both partitions, then each head band is on a separate carrier of the given length.
<code>Width</code> ?	float	Width of the head bands and carrier.
<code>Glue</code> *	element	The carrier can be applied to the book block with glue. The coordinate system for the Glue is defined in ▶ Section 6.33 EndSheetGluingParams.

6.40 HoleMakingParams

HoleMakingParams specifies the shape and positions of holes in a *Component*.

Resource Properties

Intent Pairing: *HoleMakingIntent*

Input of Processes: *HoleMaking*

Table 6.79: *HoleMakingParams* Resource

NAME	DATA TYPE	DESCRIPTION
<i>HolePattern</i> +	element	Description of individual or lines of <i>HolePattern</i> elements.

6.41 ImageCompressionParams

ImageCompressionParams provides a set of controls that determines how images will be compressed.

Resource Properties

Input of Processes: *PDLCreation*

Table 6.80: *ImageCompressionParams* Resource

NAME	DATA TYPE	DESCRIPTION
<i>ImageCompression</i> +	element	Specifies how images SHALL be compressed. If multiple <i>ImageCompression</i> elements apply to the same images, the compressions SHALL be applied in sequential order.

6.42 ImageEnhancementParams

ImageEnhancementParams describes the controls for manipulating images.

Resource Properties

Input of Processes: *ImageEnhancement*

Table 6.81: *ImageEnhancementParams* Resource

NAME	DATA TYPE	DESCRIPTION
<i>ImageEnhancementOp</i> +	element	Each <i>ImageEnhancementOp</i> describes an individual enhancement operation. The XML order of <i>ImageEnhancementOp</i> elements is significant. Multiple elements that apply to the same object SHALL be applied in that XML order.

6.42.1 ImageEnhancementOp

Table 6.82: *ImageEnhancementOp* Element

NAME	DATA TYPE	DESCRIPTION
<i>Operation</i>	NMTOKEN	Individual enhancement operation name. Values include: <i>BestGuess</i> – Best guess automated improvements based on image analysis. <i>Blurring</i> – Image blurring. <i>RedEyeRemoval</i> – Automated removal of red eye artifacts in images. <i>Sharpening</i> – Image sharpening.
<i>OperationDetails</i> ?	string	Additional details of the <i>@Operation</i> . The values are implementation specific.
<i>SourceObjects</i> ?	enumerations	Identifies which class(es) of incoming graphical objects SHALL be operated on. If <i>@SourceObjects</i> is not specified then <i>ImageEnhancementOp</i> SHALL apply to all object classes. Allowed values are from: ▶ <i>SourceObjects</i> .

6.43 ImageSetterParams

ImageSetterParams specifies the settings for an imagesetter. Both filmsetter settings and platesetter settings are described with this resource.

Resource Properties

Input of Processes: **ImageSetting**

Table 6.83: ImageSetterParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>AdvanceDistance</i> ?	float	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	Specifies the axis around which a device SHALL center an image. Allowed value is from: ▶ Axis.
<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	This attribute specifies the axis around which a device SHALL mirror an image. Allowed value is from: ▶ Axis.
<i>Polarity</i> ?	enumeration	Definition of the polarity of the image. Allowed value is from: ▶ Polarity.
<i>RollCut</i> ?	float	Length of media to be cut off of a roll, in points.

6.44 Ink

Ink describes the ink, primer, toner or varnish that is applied to a substrate when printing or varnishing. Whereas *Color* describes the visual properties of a colorant, *Ink* describes the physical material that is applied to the substrate. The default unit of measurement for *Ink* is *@Unit* = "g" (gram).

Resource Properties

Intent Pairing: **ColorIntent**Input of Processes: **ConventionalPrinting, DigitalPrinting, Varnishing**

Table 6.84: Ink Resource

NAME	DATA TYPE	DESCRIPTION
<i>InkType</i> ?	NMTOKENS	<i>@InkType</i> SHALL list specific ink type and qualities, e.g. <i>@InkType</i> ="Gloss Relief Varnish". Values include those from: ▶ Ink and Varnish Coatings.
<i>SpecificYield</i> ?	float	Weight per area at total coverage in g/m ² .
<i>IdentificationField</i> *	element	<i>IdentificationField</i> associates bar codes or labels with the container of this <i>Ink</i> .

6.45 InkZoneCalculationParams

InkZoneCalculationParams specifies the parameters for the *InkZoneCalculation* process.

Resource Properties

Input of Processes: **InkZoneCalculation**

Table 6.85: InkZoneCalculationParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>PrintableArea</i> ?	rectangle	Position and size of the printable area of the print cylinder in the coordinates of the <i>Preview</i> resource.
<i>ZoneHeight</i> ?	float	The width of one zone in the feed direction of the printing machine being used.

Table 6.85: 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 ?	float	The width of one zone of the printing machine being used. The width of a zone SHOULD be the width of an ink slide.
Device ?	element	Device provides a reference to the press that the InkZoneProfile is defined for and MAY be used to gather information on ink zone geometry.

6.46 InkZoneProfile

InkZoneProfile 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

Input of Processes: **ConventionalPrinting**

Output of Processes: **InkZoneCalculation**

Table 6.86: InkZoneProfile Resource

NAME	DATA TYPE	DESCRIPTION
ZoneHeight ?	float	The width of one zone in the feed direction of the printing machine being used.
ZoneSettingsX	FloatList	Each entry of the @ZoneSettingsX attribute is the value of one ink zone in the X 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 SHALL specify no ink and 1 SHALL specify 100% coverage.
ZoneSettingsY ?	FloatList	Each entry of the @ZoneSettingsY attribute is the value of one ink zone in the 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 SHALL specify no ink and 1 SHALL specify 100% coverage.
ZoneWidth	float	The width of one zone of the printing machine being used. Typically, the width of a zone is the width of an ink slide.

6.47 InsertingParams

InsertingParams specifies the parameters for the **Inserting** process. ▶ Table 6.88 Location of Inserts shows the various components involved in an inserting process, and how they interact.

Resource Properties

Intent Pairing: **AssemblingIntent, BindingIntent**

Input of Processes: **Inserting**

Table 6.87: InsertingParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
FinishedPage ?	integer	Finished page number of the mother Component on which the child Component SHALL be placed. @FinishedPage SHALL NOT be specified unless @InsertLocation="FinishedPage" . Corresponds to AssemblingIntent /*/@Folio.
InsertLocation	enumeration	Where to place the “child” sheet. Allowed values are: Back FinishedPage – Place the child exactly onto the page specified in @FinishedPage . Front Overfold – Place onto the overfold side.



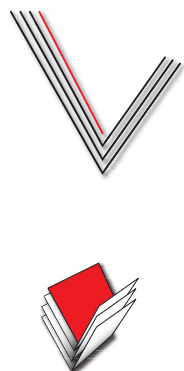
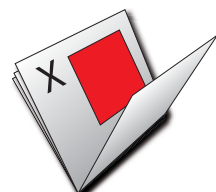
Table 6.87: InsertingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Method</i> ?	enumeration	Inserting method. Allowed values are: <i>BindIn</i> – Apply glue to fasten the insert. <i>BlowIn</i> – Loose insert.
<i>Glue</i> *	element	Array of all <i>Glue</i> elements. The coordinate system is defined by the mother <i>Component</i> .

Location of Inserts

The following graphics depict the various values of *InsertingParams*/*@InsertLocation*:

Table 6.88: 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 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.

6.48 InterpretingParams

InterpretingParams contains the parameters needed to interpret PDL pages. *InterpretingParams* itself is a generic resource that contains attributes that are relevant to all PDLs. PDL-specific details resources MAY be included as subelements of this generic resource. This specification defines one additional PDL-specific resource instance:

PDFInterpretingParams.

Resource Properties

Input of Processes: **Interpreting**

Table 6.89: InterpretingParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Center</i> ?	boolean	Indicates whether or not the finished page image SHALL be centered within the imagable area of the media. <i>@Center</i> is ignored if <i>FitPolicy</i> / <i>@SizePolicy</i> ="ClipToMaxPage" and clipping is specified.
<i>FilmRef</i> ?	IDREF	Reference to film <i>Media</i> . This <i>Resource</i> provides a description of the physical media that will be marked. The physical characteristics of the media MAY affect decisions made during <i>Interpreting</i> .

Table 6.89: InterpretingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>MirrorAround</i> ?	enumeration	This attribute specifies the axis around which a RIP SHALL mirror an image. Note: This is mirroring in the RIP and not in the hardware of the output device. Allowed value is from: ▶ Axis.
<i>PaperRef</i> ?	IDREF	Reference to final paper <i>Media</i> . This <i>Resource</i> provides a description of the physical media that will be marked. The physical characteristics of the media MAY affect decisions made during <i>Interpreting</i> .
<i>PlateRef</i> ?	IDREF	Reference to plate <i>Media</i> . This <i>Resource</i> provides a description of the physical media that will be marked. The physical characteristics of the media MAY affect decisions made during <i>Interpreting</i> .
<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 value is from: ▶ Polarity.
<i>PrintQuality</i> ?	enumeration	Generic switch for setting the quality of an otherwise inaccessible device. Allowed values are: High – Highest quality available on the printer. Normal – The default quality provided by the printer. Draft – Lowest quality available on the printer.
<i>ProofPaperRef</i> ?	IDREF	Reference to paper <i>Media</i> used for proofing. This <i>Resource</i> provides a description of the physical media that will be marked. The physical characteristics of the media MAY affect decisions made during <i>Interpreting</i> .
<i>Scaling</i> ?	XYPair	A pair of positive real values that indicates the scaling factor for the content. Values between 0 and 1 specify that the contents SHALL be reduced, while values greater than 1 specify that the contents SHALL be expanded. Any scaling defined in <i>FitPolicy</i> SHALL be applied after the scaling defined by this attribute.
<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> .
<i>FitPolicy</i> ?	element	Allows printing even if the size of the imagable area of the media does not match the requirements of the data.
<i>InterpretingDetails</i> ?	element	Container for interpreter-specific details.
<i>ObjectResolution</i> *	element	Indicates the resolution at which the PDL contents will be interpreted in DPI. These elements MAY be different from the <i>ObjectResolution</i> elements provided in <i>RenderingParams</i> .
<i>PDFInterpretingParams</i> ?	element	Details of interpreting for PDF.

6.48.1 InterpretingDetails

InterpretingDetails contains PDL-specific instructions for an interpreter.

Table 6.90: InterpretingDetails Element

NAME	DATA TYPE	DESCRIPTION
<i>MinLineWidth</i> ?	float	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.

6.48.2 PDFInterpretingParams

Table 6.91: PDFInterpretingParams Element

NAME	DATA TYPE	DESCRIPTION
<i>EmitPDFBG</i> ?	boolean	Indicates whether BlackGeneration functions SHALL be emitted.
<i>EmitPDFHalftones</i> ?	boolean	Indicates whether halftones SHALL be emitted.
<i>EmitPDFTransfers</i> ?	boolean	Indicates whether transfer functions SHALL be emitted.
<i>EmitPDFUCR</i> ?	boolean	Indicates whether under color removal functions SHALL be emitted.
<i>HonorPDFOverprint</i> ?	boolean	Indicates whether or not overprint settings in the file SHALL be honored. If "true", the setting for overprint SHALL be honored. If "false", it is expected that the device does not directly support overprint and that the PDF is pre-processed to simulate the effect of the overprint settings.
<i>ICCColorAsDeviceColor</i> ?	boolean	Indicates whether colors specified by ICC color spaces are to be treated as device colorants.
<i>OCGIntent</i> ?	NMTOKEN	The value of <i>@OCGIntent</i> sets the intent for which OCGs SHALL be selected. Values include: <i>Design</i> – As described in ▶ [PDF1.6]. <i>View</i> – As described in ▶ [PDF1.6].
<i>OCGProcess</i> ?	NMTOKEN	The value of <i>@OCGProcess</i> sets the purpose for which the Interpreting process is being performed. This, in turn, sets which value from a relevant optional content usage dictionary SHALL be used to determine whether each OCG is included in the output. Values include: <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. Additional values are defined in ▶ [ISO19593-1:2016] or MAY be site specific.
<i>OCGZoom</i> ?	float	The value of <i>@OCGZoom</i> sets the magnification that SHALL be assumed in comparisons with the zoom dictionary in a relevant optional content usage dictionary to determine whether each OCG is included in the <i>RunList</i> . A <i>@OCGZoom</i> value of 1.0 corresponds 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 <i>@PrintTrapAnnotations</i> .
<i>PrintTrapAnnotations</i> ?	boolean	Indicates whether the contents of trap annotations on PDF pages SHALL be included in the output.
<i>TransparencyRenderingQuality</i> ?	float	Values are 0 to 1. A value of 0 represents the lowest allowable quality; 1 represents the highest achievable quality.
<i>OCGControl</i> *	element	<i>OCGControl</i> provides a list of the OCGs (layers) that SHALL be explicitly included or excluded in the <i>RunList</i> . Any OCGs not listed in an <i>OCGControl</i> element SHALL follow the rules set by the referenced PDF.
<i>ReferenceXObjectParams</i> ?	element	Describes how the interpreter should handle PDF Reference XObjects.

6.48.3 ReferenceXObjectParams

Table 6.92: ReferenceXObjectParams Element

NAME	DATA TYPE	DESCRIPTION
<i>Mode</i>	NMTOKEN	Specifies how to handle a Reference XObject's reference. Values include: Ignore – The reference SHALL be ignored, and no content is imaged for that Reference XObject. If proxy content is supplied with the Reference XObject, it SHALL be imaged. ResolveAlways – An attempt SHALL be made to resolve the reference and image the graphics described by that reference. ResolveIfPDFX5 – An attempt SHALL be made to resolve the reference ONLY if the PDF file is a valid PDF/X-5 file, AND the referenced file passes the criteria required by PDF/X-5, see ▶ [ISO15930-8:2010].
FileSpec (<i>SearchPath</i>) *	element	An ordered list of FileSpec 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.

6.49 JacketingParams

Description of the setup of the jacketing machinery. Jacket height and width (1 and 4 in the ▶ Figure 6-22: Setup of the jacketing machinery) are specified within the **Component** that describes the jacket.

Resource Properties

Intent Pairing: **BindingIntent**

Input of Processes: **Jacketing**

Table 6.93: JacketingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>FoldingDistance</i> ?	float	Distance from the fold at <i>@FoldingWidth</i> to the other fold. If not specified, it defaults to the width of the jacket minus double the value of <i>@FoldingWidth</i> (symmetrical folds).
<i>FoldingWidth</i>	float	Definition of the dimension of the folding width of the front cover fold, see ▶ Figure 6-23: Parameters and coordinate system for jacketing. All other measurements are implied by the dimensions of the book.

Figure 6-22: Setup of the jacketing machinery

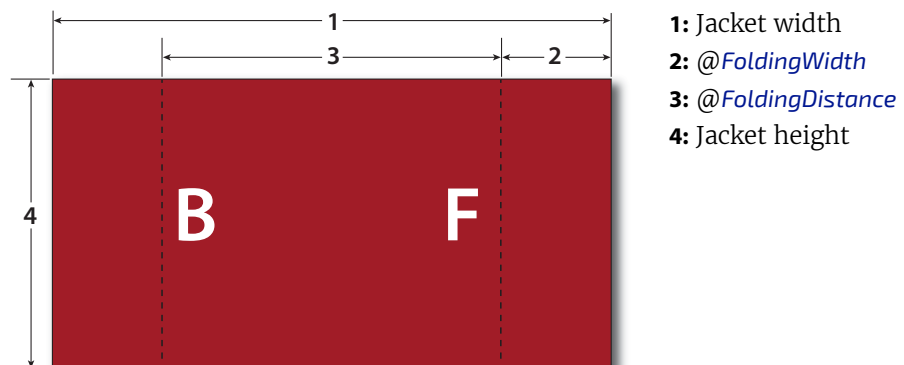
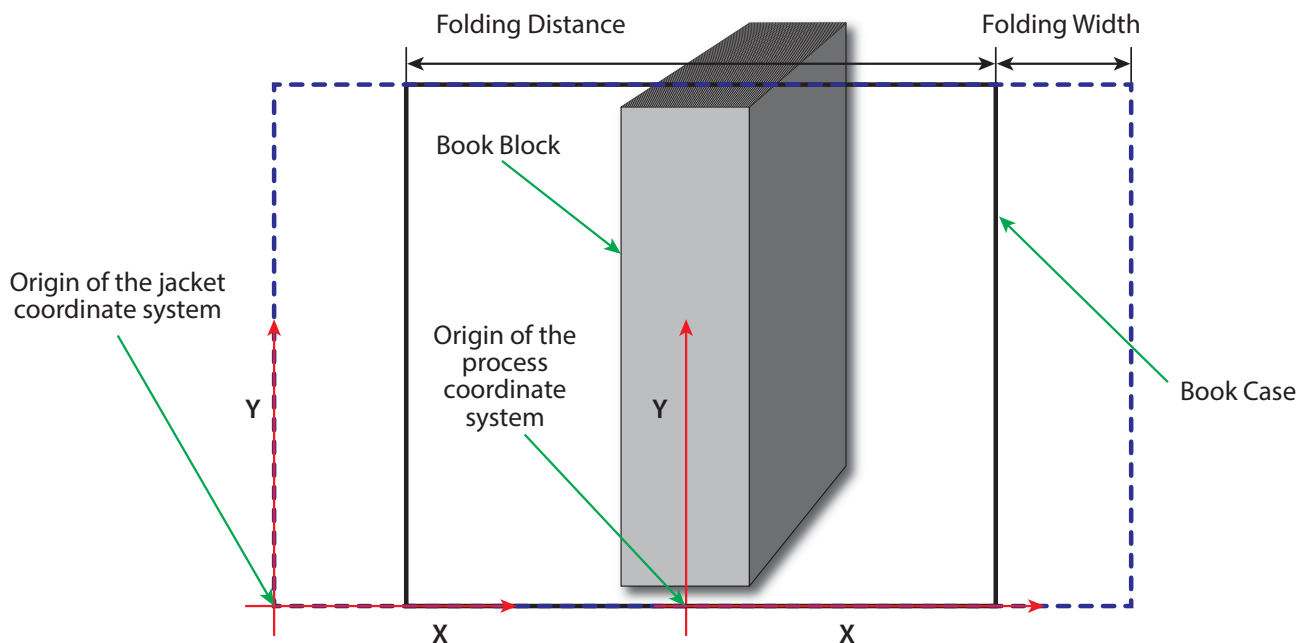


Figure 6-23: Parameters and coordinate system for jacketing



6.50 LabelingParams

LabelingParams defines the details of the **Labeling** process.

Resource Properties

Input of Processes: **Labeling**

Table 6.94: LabelingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>Application</i> ?	NMTOKEN	Application method of the label. Values include: Glue – Glued onto the component. Loose – Loosely laid onto the component. SelfAdhesive – Self adhesive label. Staple – Stapled onto the component.
<i>Face</i> ?	enumeration	Position of the label on the bundle. Allowed value is from: ▶ Face.
<i>Offset</i> ?	XYPair	Position of the lower left corner of the label after applying the rotation defined in Resource/@Orientation of the corresponding Component(Label) relative to the lower left corner of the component surface as defined by @Face .
<i>FileSpec</i> ?	element	A FileSpec element pointing to an address list. The format of the referenced mailing list is implementation dependent.

6.51 LaminatingParams

LaminatingParams specifies the parameters needed for laminating.

Resource Properties

Intent Pairing: **LaminatingIntent**

Input of Processes: **Laminating**

Table 6.95: LaminatingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>GapList</i> ?	FloatList	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>LaminatingBox</i> ?	rectangle	Area on the Component that SHALL be laminated.
<i>LaminatingMethod</i> ?	enumeration	Laminating technology that SHALL be applied. Allowed values are: CompoundFoil DispersionGlue Fusing
<i>ModuleID</i> ?	NMTOKEN	Identifier of the laminating module in a multi-function device, such as a printing press. See Module for details.
<i>NipWidth</i> ?	float	Width of the nip in points that SHALL be formed between the fusing rollers and the component in the Laminating process.
<i>Temperature</i> ?	float	Temperature that SHALL be used in the Laminating process.

6.52 Layout

Layout is used both for fixed-layout and for automated printing.

Layout MAY be used to describe the exact details of a press sheet that is processed by an imposition engine to place marks and pages. The **Layout** resource is also a high-level description of how a **Component** SHALL be produced. This high level representation is typically produced by the MIS production planning module and consumed by a prepress workflow system.

Layout MAY specify how the surfaces of the **BinderySignature** elements of a job are placed onto press sheets.

Resource Properties

Intent Pairing: **LayoutIntent**

Input of Processes: **ConventionalPrinting, DigitalPrinting, Imposition, InkZoneCalculation, Stripping**

Output of Processes: **Stripping**

Table 6.96: Layout Resource (Sheet 1 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>Automated</i> ?	boolean	If "true", the Imposition process is expected to perform automated imposition. <i>Layout/@Automated</i> SHALL be identical for all Layout elements of a ResourceSet .
<i>FilmRef</i> ?	IDREF	Reference to film Media that SHALL provide a description of the physical media that will be marked.
<i>InnermostShingling</i> ?	float	Relative creep compensation that SHALL be applied to the innermost part of the assembled booklet. A value of "1" SHALL specify 100% creep compensation. Shingling is perpendicular to the spine. Negative values go towards the spine. Values for pages between inner and outer SHALL be interpolated. Actual values of shingling are calculated by the system or operator. See ▶ Figure 6-24: Shingling for stripping and ▶ Figure 6-25: Shingling for stripping – details.

Table 6.96: Layout Resource (Sheet 2 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>MaxCollect</i> ?	integer	Maximum number of sheets that SHALL be collected into a signature. <i>@MaxCollect</i> modifies the pagination when automated imposition is selected by splitting documents of the input <i>RunList</i> into smaller virtual documents that fill exactly one signature. Thus the ord counting within a <i>Layout</i> SHALL be restarted at 0 after <i>@MaxCollect</i> sheets. The distribution of signatures with less than <i>@MaxCollect</i> sheets within the final product is implementation dependent. Example: If 10 sheets are required and <i>@MaxCollect</i> =4, then 3-3-4, 4-4-2 or any permutations are possible. See also <i>@MinCollect</i> . If not specified, all sheets SHALL be collected into one signature.
<i>MinCollect</i> ?	integer	Minimum number of sheets that SHALL be collected into a signature. <i>@MinCollect</i> modifies the pagination when automated imposition is selected. The distribution of signatures with less than <i>@MaxCollect</i> sheets within the final product is implementation dependent. Example: If 10 sheets are required and <i>@MaxCollect</i> =4, <i>@MinCollect</i> =3, then only 3-3-4, or any permutations are possible. Example: If 2 sheets are required and <i>@MinCollect</i> =3, then a third sheet with only blank pages SHALL be produced.
<i>OutermostShingling</i> ?	float	Relative creep compensation that SHALL be applied to the outermost part of the assembled booklet. A value of "1" SHALL specify 100% creep compensation. Shingling is perpendicular to the spine. Negative values go towards the spine. Values for pages between inner and outer SHALL be interpolated. Actual values of shingling are calculated by the system or operator. See ▶ Figure 6-24: Shingling for stripping and ▶ Figure 6-25: Shingling for stripping – details.
<i>PaperRef</i> ?	IDREF	Reference to final paper <i>Media</i> that SHALL provide a description of the physical media that will be marked.
<i>PlateRef</i> ?	IDREF	Reference to plate <i>Media</i> that SHALL provide a description of the physical media that will be marked.
<i>ProofPaperRef</i> ?	IDREF	Reference to paper <i>Media</i> that SHALL be used for proofing and that SHALL provide a description of the physical media that will be marked.
<i>SurfaceContentsBox</i> ?	rectangle	This box, specified in <i>Layout</i> -coordinate space, defines the area into which <i>PlacedObject</i> elements SHALL be positioned. Content that is outside of the area specified by <i>@SurfaceContentsBox</i> SHALL be clipped. The lower left corner of the rectangle establishes the coordinate system onto which the content SHALL be positioned and SHOULD have a value of "0 0". If this attribute is not supplied, the origin SHALL be "0 0" and the extent is undefined.
<i>WorkStyle</i> ?	enumeration	The direction in which to turn the press sheet. Allowed value is from: ▶ <i>WorkStyle</i> .
<i>Device</i> *	element	List of <i>Device</i> resources that the MIS expects to execute this <i>Layout</i> . This MAY include prepress devices, presses or finishing devices.
<i>FileSpec</i> (ExternalImpositionTemplate) ?	element	Reference to an external imposition template in a proprietary format. <i>Layout</i> SHOULD NOT contain information that overlaps information specified in <i>FileSpec</i> (<i>ExternalImpositionTemplate</i>). Information specified in <i>Layout</i> SHALL override parameters specified in <i>FileSpec</i> (<i>ExternalImpositionTemplate</i>).
<i>FitPolicy</i> ?	element	Specifies automated fit policy for content that is placed onto the surface that is specified in <i>@SurfaceContentsBox</i> .
<i>PlacedObject</i> *	element	<i>PlacedObject</i> elements specify content or marks that SHALL be placed on the surface in the order in which they occur in the <i>Layout</i> . All <i>PlacedObject</i> elements SHALL be specified in <i>Layout Resources</i> that contain at least one <i>Part</i> / <i>@Side</i> .

Table 6.96: Layout Resource (Sheet 3 of 3)

NAME	DATA TYPE	DESCRIPTION
Position *	element	The Position elements specify how the BinderySignature SHALL be placed onto a sheet. Multiple Position objects that reference the same BinderySignature specify multiple identical BinderySignature elements with the same content. If Position elements are specified and @WorkStyle="WorkAndTurn" or @WorksStyle="WorkAndTumble" the mirrored Position elements are not implied and Position elements SHALL be specified for those positions.
SheetActivation ?	element	Specifies the conditions under which the optional sheet defined by this Layout SHALL be produced. SheetActivation SHALL only be present when Layout/@Automated="true" . SheetActivation SHALL activate the surface as specified by Part/@Side if present. Thus if two surfaces are present in the Layout and only one surface contains SheetActivation , then the sheet shall be printed on one side only. If Part/@Side is not specified, e.g. if the Layout is used as an input to Stripping , then SheetActivation SHALL apply to both surfaces of the sheet.
StripMark *	element	StripMark provides a description of production marks for Stripping in the context of the Layout . See also PlacedObject/MarkObject for a description of the individual marks.

Figure 6-24: Shingling for stripping

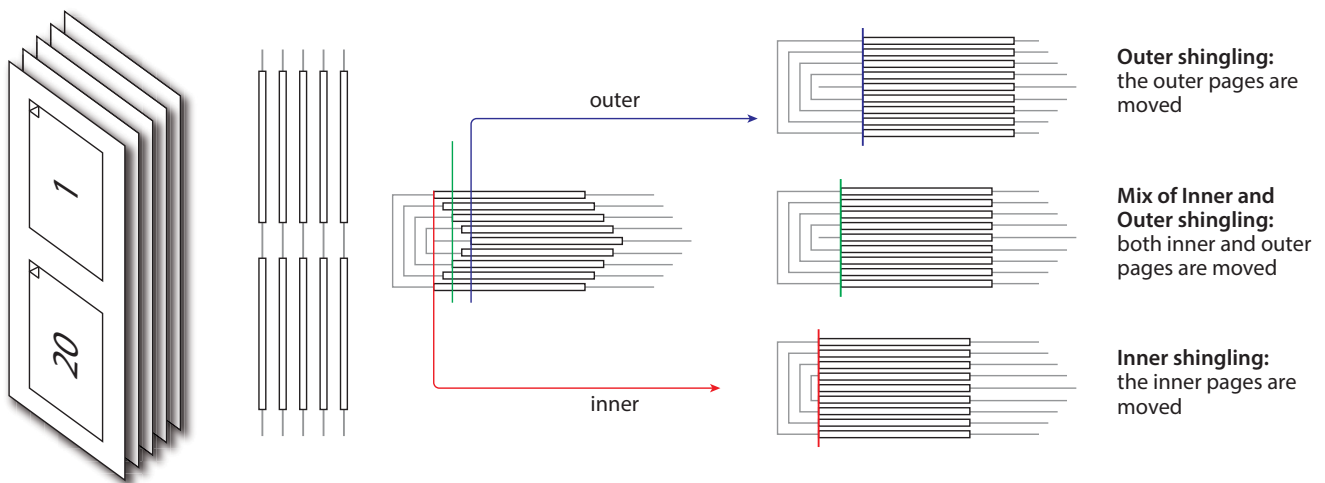
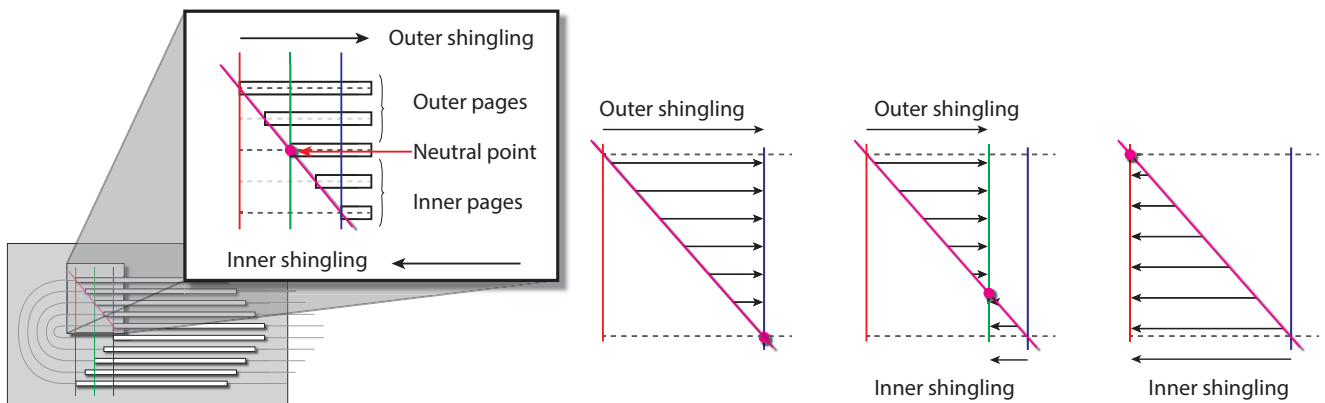


Figure 6-25: Shingling for stripping – details



6.52.1 CIELABMeasuringField

Information about a color measuring field. The color is specified as a CIE-L*a*b* value.

Table 6.97: 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> ?	float	Diameter of the measuring field.
<i>Percentages</i> ?	FloatList	Percentage values for each separation. The number of array elements SHALL match the number of separations.
<i>ScreenRuling</i> ?	FloatList	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> ?	float	Tolerance in ΔE.

6.52.2 ColorControlStrip

ColorControlStrip 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. Its coordinates (x_0 , y_0) 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

φ = Value of the @Rotation attribute

Table 6.98: ColorControlStrip Element

NAME	DATA TYPE	DESCRIPTION
<i>Center</i> ?	XYPair	Position of the center of the color control strip in the coordinates of the MarkObject that contains this mark.
<i>Rotation</i> ?	float	Rotation in degrees. Positive graduation figures indicate counter-clockwise rotation; negative figures indicate clockwise rotation.
<i>Separations</i> ?	NMTOKENS	Ordered list of separation identifiers that comprise the ColorControlStrip . If neither CIELABMeasuringField nor DensityMeasuringField are specified, the geometry is implied by the value of @StripType. Additional details of the colorants SHOULD be provided in ResourceSet [@Name="Color"].
<i>Size</i> ?	XYPair	Size, in points, of the color control strip.
<i>StripType</i> ?	string	Type of color control strip. This attribute MAY be used for specifying a pre-defined, 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 .

6.52.3 Condition

The condition element defines the condition when a *PageActivation*, *PageCondition* or *SheetActivation* is active when processing a layout with *@Automated="true"*.

Table 6.99: Condition Element

NAME	DATA TYPE	DESCRIPTION
<i>PartContext</i> ?	NMTOKENS	List of attribute names within <i>Part</i> that SHALL reset the context of the <i>Part</i> elements in this <i>Condition</i> . The <i>Part</i> elements in this <i>Condition</i> SHALL NOT contain any attributes that are in <i>@PartContext</i> . Example: <i>@PartContext="DocIndex"</i> and <i>Part/@RunIndex="01"</i> . In this case <i>@RunIndex</i> is recalculated whenever <i>@DocIndex</i> evaluates to a different value. Note: Without <i>@PartContext</i> and no explicit <i>Part/@DocIndex</i> ; <i>@RunIndex</i> would be evaluated in the context of the entire <i>RunList</i> .
<i>Part</i> +	element	This <i>Condition</i> SHALL evaluate to "true" whenever any <i>RunList</i> partition matches at least one of the <i>Part</i> elements.

6.52.4 ContentObject

ContentObject elements identify containers for page content on a surface. They SHALL be filled from the *RunList(Document)* of the *Imposition* or *Stripping* process. *ContentObject* SHALL be an empty element.

6.52.5 CutMark

CutMark 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 6.100: CutMark Element

NAME	DATA TYPE	DESCRIPTION
<i>MarkType</i>	enumeration	Cut mark type. Allowed values are from: ▶ Table 6.101 MarkType Attribute Values.
<i>Position</i>	XYPair	Position of the logical center of the cut mark in the coordinates of the <i>MarkObject</i> that contains this mark. Note: The logical center of the cut mark does not always coincide with the center of the visible cut mark symbol.

Table 6.101: MarkType Attribute Values (Sheet 1 of 2)










VALUE	SYMBOL	DESCRIPTION
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.

Table 6.101: MarkType Attribute Values (Sheet 2 of 2)

VALUE	SYMBOL	DESCRIPTION
LowerLeftCutMark		Corner at logical position.
UpperLeftCutMark		Corner at logical position.
LowerRightCutMark		Corner at logical position.
UpperRightCutMark		Corner at logical position.

6.52.6 DensityMeasuringField

DensityMeasuringField contains information about a density measuring field.

Table 6.102: DensityMeasuringField Element

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 , @Center SHALL refer to the rectangle defined by @Center and @Size of the ColorControlStrip .
<i>Density</i>	FloatList	Density value for each process color measured with filter. The sequence of colors SHALL be C M Y K.
<i>Diameter</i>	float	Diameter of the density measuring field.
<i>DotGain</i>	float	Percentage of dot gain.
<i>Separation</i>	NMTOKEN	Separation identifier that this DensityMeasuringField applies to. Additional details of the colorants SHOULD be provided in ResourceSet [@Name="Color"].
<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).

6.52.7 MarkObject

MarkObject elements identify containers for production marks on a surface. If the containing **Layout** is used as an input to **Imposition**, then the PDL SHALL exist and the marks SHALL be filled from the **RunList** (Marks) of the **Imposition** process. If the containing **Layout** is used as an input to **Stripping**, the imposition engine SHALL generate the marks on the fly. The content data in individual **MarkObject** elements MAY contain multiple logical marks.

Table 6.103: MarkObject Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ContentRef</i> ?	IDREF	@ContentRef refers to the ContentObject that shall define the scope of the RunList metadata that is used to evaluate Condition elements in this MarkObject .
<i>ColorControlStrip</i> *	element	ColorControlStrip describes a color control strip.

Table 6.103: MarkObject Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
CutMark *	element	CutMark describes cut marks on a sheet.
RegisterMark *	element	RegisterMark describes a register mark that can be used to measure color registration.
ScavengerArea *	element	ScavengerArea describes a scavenger area for removing excess ink from printed sheets.

6.52.8 PageActivation

PageActivation SHALL define when content SHALL be conditionally placed in its parent **PlacedObject**. Before placing page content from a **RunList** into a **PlacedObject**, **PageActivation/Condition** SHALL be evaluated in the context of the current partition of the **RunList**.

Table 6.104: PageActivation Element

NAME	DATA TYPE	DESCRIPTION
Condition +	element	This PageActivation SHALL be applied when all Condition elements evaluate to true.

6.52.9 PageCondition

The **PageCondition** element defines restrictions on when content SHALL NOT be placed in its parent **PlacedObject**. Before placing content from a **RunList** into a **PlacedObject** the **PageCondition/Condition** elements SHALL be evaluated in the context of the current partition of the **RunList**. If the result of the evaluation is "true", this **PlacedObject** SHALL be skipped and the current content SHALL be placed into the location defined by the next **PlacedObject** that consumes the same **RunList**. This corresponds to incrementing the effective **@Ord** value of the page and all following pages in the **RunList** by 1, effectively incrementing the total number of pages of the **RunList**. If the next **PlacedObject** is also restricted then the process SHALL be repeated.

Table 6.105: PageCondition Element

NAME	DATA TYPE	DESCRIPTION
Condition +	element	This PageCondition SHALL be applied when all Condition elements evaluate to true.

Example 6.8: PageCondition

The following example shows how **PageCondition** can be used to force a chapter boundary to be on a specific page side. **PlacedObject/PageCondition/Condition** contains a **Part** element that evaluates to true whenever **@RunIndex** is 0, i.e. the first page in the context of each chapter as specified by **@PartContext**.

Note: Chapters are represented as documents in this example.

```
<ResourceSet Name="Layout" Usage="Input">
  <Resource>
    <Part Side="Front"/>
    <Layout Automated="true">
      <PlacedObject CTM="1 0 0 1 0 0" Ord="0">
        <ContentObject/>
      </PlacedObject>
      <PlacedObject CTM="1 0 0 1 500 0" Ord="1">
        <ContentObject/>
        <PageCondition>
          <Condition PartContext="DocIndex">
            <Part RunIndex="0 0"/>
          </Condition>
        </PageCondition>
      </PlacedObject>
    </Layout>
  </Resource>
</ResourceSet>
```

6.52.10 PlacedObject

PlacedObject elements describe any kind of marks or content on a surface. They SHALL contain exactly one of **ContentObject** or **MarkObject** that describes additional details of the page content or production mark that the **PlacedObject** represents.

Table 6.106: PlacedObject Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Anchor</i> ?	enumeration	Specifies the anchor point of the PlacedObject that SHALL remain in place on the surface when the value of @TrimSize changes. @Anchor is specified in the coordinate system of the PlacedObject prior to application of @CTM . Note: The @Anchor 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 a Layout resource is used as a template for that utility. @Anchor SHALL have no effect on processing when a Layout resource is input to the Imposition process. Allowed value is from: ▶ Anchor .
<i>ClipBox</i> ?	rectangle	Clipping rectangle in terms of the coordinates of the @SurfaceContentsBox .
<i>ClipPath</i> ?	PDFPath	Clip path for the PlacedObject in the coordinates of the @SurfaceContentsBox (lower left of @SurfaceContentsBox is used as reference zero point, same as for @ClipBox). The actual clip region is the intersection of @ClipBox and @ClipPath , or the intersection of @ClipBox and @SourceClipPath . In both cases two regions are applied sequentially and the resulting clip region is smaller than either of those regions. @ClipPath and @SourceClipPath SHALL NOT be specified in the same PlacedObject . @ClipPath SHOULD be specified when both @ClipPath and @SourceClipPath are known because @ClipPath provides a more stable coordinate system (not sensitive to shifts caused by editing the page).
<i>CTM</i>	matrix	The coordinate transformation matrix (CTM — a Postscript term) of the object in the @SurfaceContentsBox . 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 @SurfaceContentsBox . For details, see ▶ Section 2.6.1.1 Source Coordinate Systems. Note: @CTM 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 PlacedObject . Specified in the coordinate system of @SurfaceContentsBox .
<i>ID</i> ?	ID	Identifier for referencing this PlacedObject .
<i>Ord</i> ?	integer	@Ord SHALL specify a zero-based reference to an index in the RunList that is selected by the presence of either ContentObject or MarkObject . The index SHALL be incremented for every page in the referenced PDL. The @Ord value of the first page of a RunList SHALL have the value "0". If Layout/@Automated="true" , @Ord MAY be a negative integer. In this case, the explicit @Ord for each iteration of the automated Layout SHALL be calculated by subtracting the appropriate number of @Ord values from the back of the document or the position in the document that is calculated from Layout/@MaxCollect and Layout/@MinCollect . For details on automated Layout , see ▶ Section 5.4.8 Imposition.
<i>PositionRef</i> ?	IDREF	Reference to the Position that defines where this PlacedObject SHALL be located.

Table 6.106: PlacedObject Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>SourceClipPath</i> ?	PDFPath	<p>Clip path for the PlacedObject in the source coordinate system. @SourceClipPath 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 @SourceClipPath.</p> <p>@ClipPath and @SourceClipPath SHALL NOT be specified in the same PlacedObject.</p> <p>See @ClipPath for more details.</p> <p>See ▶ Section 2.6.1.1 Source Coordinate Systems for definitions of source coordinate systems.</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 a Layout resource is used as a template for that utility. @TrimCTM SHALL have no effect on processing when a Layout resource is input to the Imposition process.</p>
<i>TrimSize</i> ?	XYPair	<p>The size of the object's trim box as viewed in the object source coordinates (@TrimCTM scaling and rotation NOT applied).</p> <p>@TrimSize 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, @TrimSize specifies the bounding box that SHALL be used for scaling and rotation when processing Layout/FitPolicy.</p> <p>Note: Recalculation of PlacedObject/@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, PlacedObject/@CTM recalculations SHOULD always be done according to PlacedObject/@TrimCTM and PlacedObject/@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.</p> <p>Usage of both PlacedObject elements @TrimCTM and @TrimSize attributes will allow page replacements on any type of imposition Layout.</p>
<i>ContentObject</i> ?	element	<p>If a ContentObject is present, this PlacedObject shall be filled from RunList(Document) if it is used in the context of an Imposition or Stripping process. Exactly one of either ContentObject or MarkObject SHALL be present.</p>
<i>MarkObject</i> ?	element	<p>Details of the production mark. Exactly one of either ContentObject or MarkObject SHALL be present.</p>
<i>PageActivation</i> ?	element	<p>PageActivation defines conditions on which page content SHALL be placed in the parent PlacedObject. If PageActivation is present, the page SHALL NOT be filled into that PlacedObject unless the RunList matches a partition specified in any of the PageActivation/Part elements. PageActivation SHALL only be present when Layout/@Automated="true".</p>
<i>PageCondition</i> ?	element	<p>PageCondition defines conditions on which page content SHALL NOT be placed in the parent PlacedObject. If PageCondition is present, the page SHALL NOT be filled into that PlacedObject if the RunList matches a partition specified in any of the PageCondition/Part elements. PageCondition SHALL only be present when Layout/@Automated="true".</p>

6.52.11 Position

The **Position** element allows the aligned placement of a **BinderySignature** onto a **Layout**.

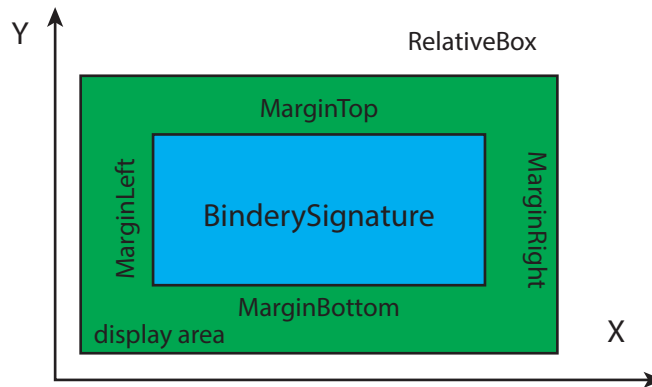
Table 6.107: Position Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>AbsoluteBox</i> ?	rectangle	<p>Absolute position, in points, of the display area of this BinderySignature on the front side of the Layout.</p> <p>The BinderySignature is placed onto the display area after applying the @Orientation transformation.</p> <p>The display area SHALL include the absolute margins defined by @MarginTop, @MarginBottom, @MarginLeft and @MarginRight. @AbsoluteBox overrides @RelativeBox if both are specified.</p> <p>If @AbsoluteBox is specified, it SHALL be used as is for all imposition calculations.</p>
<i>BinderySignatureID</i> ?	NMTOKEN	<p>If present, @BinderySignatureID SHALL reference a BinderySignature/../Part/@BinderySignatureID that SHALL be placed onto the layout at this position. If @BinderySignatureID is not specified, individual pages SHALL be placed at this Position.</p>
<i>BlockName</i> ?	NMTOKEN	<p>Identifies a CutBlock resulting from a Cutting process if the element specified by the Position is created by Cutting.</p>
<i>GangElementID</i> ?	NMTOKEN	<p>If present, @GangElementID SHALL reference a GangElement element that was placed in this position by SheetOptimizing.</p>
<i>ID</i> ?	ID	<p>Identifier of this Position. @ID SHALL be used to link PlacedObject elements to a Position.</p>
<i>MarginBottom</i> ?	float	<p>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 Layout.</p>
<i>MarginLeft</i> ?	float	<p>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 Layout.</p>
<i>MarginRight</i> ?	float	<p>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 Layout.</p>
<i>MarginTop</i> ?	float	<p>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 Layout.</p>
<i>Orientation</i> ?	enumeration	<p>Named orientation describing the transformation of the orientation of the BinderySignature on the Layout. For details, see ▶ Table 2.1 Matrices and Orientation values for describing the orientation of a Component.</p> <p>Allowed value is from: ▶ Orientation.</p>
<i>RelativeBox</i> ?	rectangle	<p>@RelativeBox is a rough definition of the general position of the display area of this BinderySignature on the front side of the Layout. The BinderySignature SHALL be placed onto the display area after applying the @Orientation transformation.</p> <p>The display area SHOULD include the absolute margins defined by @MarginTop, @MarginBottom, @MarginLeft and @MarginRight. @AbsoluteBox overrides @RelativeBox if both are specified.</p> <p>If neither @AbsoluteBox nor @RelativeBox are specified, the full relative media box "0 0 1.0 1.0" SHALL be applied.</p>
<i>StackDepth</i> ?	integer	<p>Maximum number of sheets on a stack for cut and stack imposition. Implementations SHOULD generate the minimum stack size to accommodate the available number of input pages. If Position/@Orientation specifies a flip of the pages, then the stack SHALL be filled in descending page order.</p>

Table 6.107: Position Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>StackOrd</i> ?	integer	Index of the stack. The stacks SHALL be processed in the sequence of ascending @ <i>StackOrd</i> . If @ <i>StackOrd</i> is specified for any individual <i>Layout</i> partition, @ <i>StackOrd</i> SHALL be specified for all <i>Position</i> elements of that <i>Layout</i> .

Figure 6-26: RelativeBox including margins



6.52.12 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.

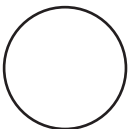
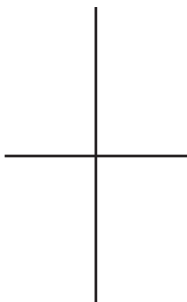
Table 6.108: RegisterMark Element

NAME	DATA TYPE	DESCRIPTION
<i>Center</i> ?	XYPair	Position of the center of the register mark in the coordinates of the <i>MarkObject</i> that contains this mark.
<i>MarkType</i> ?	NMTOKENS	Type of <i>RegisterMark</i> . Note: Marks can be combined to form a composite mark, see ▶ Figure 6-27: Combined MarkType. Values include those from: ▶ Table 6.101 MarkType Attribute Values.
<i>MarkUsage</i> ?	enumerations	Specifies the usage of the <i>RegisterMark</i> . 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.
<i>Rotation</i> ?	float	Rotation in degrees. Positive values indicate counter-clockwise rotation; negative values indicate clockwise rotation.
<i>Separations</i> ?	NMTOKENS	Set of separation identifiers to which the register mark is bound. Additional details of the colorants SHOULD be provided in <i>ResourceSet</i> [@Name="Color"].

Table 6.109: MarkType Attribute Values (Sheet 1 of 2)

VALUE	DESCRIPTION
<i>Arc</i>	

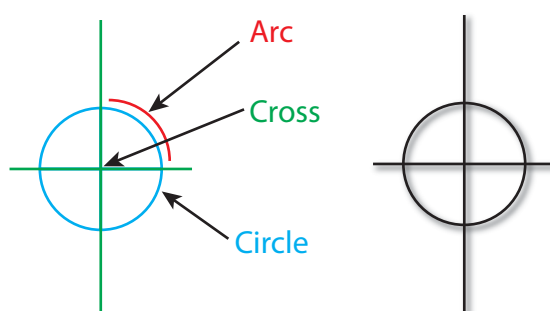
Table 6.109: MarkType Attribute Values (Sheet 2 of 2)

VALUE	DESCRIPTION
Circle	
Cross	

6.52.12.1 Combined Register Mark

The following figure illustrates a combined **RegisterMark** consisting of an arc, a cross and a circle as well as the individual mark components.

Figure 6-27: Combined MarkType



6.52.13 ScavengerArea

ScavengerArea describes a scavenger area for removing excess ink from printed sheets. It is defined within a **MarkObject** of a surface.

Table 6.110: ScavengerArea Element

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>Separations</i> ?	NMTOKENS	Set of separation identifiers to which the scavenger area is bound. Additional details of the colorants SHOULD be provided in ResourceSet [@Name="Color"].
<i>Size</i> ?	XYPair	Size of the scavenger area.

6.52.14 SheetActivation

Table 6.111: SheetActivation Element

NAME	DATA TYPE	DESCRIPTION
Condition +	element	This SheetActivation shall be applied when all Condition elements evaluate to true for any ContentObject that is supplied on the unconditional Layout partition that is currently active.

6.52.15 More about Layout

6.52.15.1 Partition Key Restrictions

If **PlacedObject** elements are placed onto a **Layout**, **Part/@Side** SHALL be specified.

All **PlacedObject** elements that are intended to be imaged on one surface SHALL be specified in one partition.

6.52.15.2 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 object** – A rectangle that is PDL-specific that indicates the area of the object that indicates the intended trimming area.

6.52.15.3 Finding the Trim Box of an Object

RunList/@SourceTrimBox always takes precedence over boxes defined inside the PDL. 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:

- EPS – **HiResBoundingBox** then **BoundingBox**
- PDF – **CropBox** then **MediaBox**

Note: These boxes might not be correct in all cases.

6.52.15.4 Using Ord to Reference Elements in RunList Resources

PlacedObject/@Ord represents a reference to a *logical* page in a **RunList**. The index SHALL be incremented for every page of the **RunList**. The reference shall be calculated in the context of the **ResourceSet**. The following examples illustrate the usage of **@Ord**.

Example 6.9: RunList: Simple Multi-File 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.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="RunList" Types="Imposition">
  <ResourceSet Name="RunList" Usage="Input">
    <Resource>
      <Part Run="R1"/>
      <RunList NPage="6" Pages="0 5">
        <FileSpec URL="File:///File1.pdf"/>
      </RunList>
    </Resource>
    <Resource>
      <Part Run="R2"/>
      <RunList Pages="0 -1">
        <FileSpec URL="File:///File2.pdf"/>
      </RunList>
    </Resource>
  </ResourceSet>
</XJDF>
```

Table 6.112: Example 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)

6.52.15.5 Calculating Ord values in automated imposition

If `Layout/@Automated="true"`, the values of `@Ord` SHALL be recalculated for every iteration of a sheet:

- **Step 1:** Calculate the number of pages that each sheet consumes. Each unique `@Ord` value consumes one page. Positive ord values consume pages from the front of the list and negative ord values consume pages from the back of the list.
- **Step 2:** Calculate the number of pages per signature by multiplying the number of pages per sheet with `Layout/@MaxCollect`. If `Layout/@MaxCollect` is not specified, the number of signatures is 1 and all pages SHALL be collected into one signature.
- **Step 3:** Iterate over all signatures. The document reference to the list of pages is incremented by the number of pages that a signature consumes.
- **Step 4:** For each signature, iterate over all sheets in the signature, while maintaining the references to the front and back of the current list of pages.

The values in the following table are valid for the following pre-existing conditions:

- Number of pages consumed per sheet is four (`@Ord=0, 1, -1, -2`)
- `Layout/@MaxCollect="3"`, therefore the number of pages consumed per signature is $3*4 = 12$
- Number of pages in the `RunList` is 20, therefore the number of sheets is $(20+(4-1))/4 = 5$

Table 6.113: Example Calculating Ord values in automated Imposition

SIGNATURE	SHEET	ORD VALUE	CALCULATED ORD	SIGNATURE	SHEET	ORD VALUE	CALCULATED ORD
0	0	0	0	1	0	0	12
0	0	1	1	1	0	1	13
0	0	-2	10	1	0	-2	18
0	0	-1	11	1	0	-1	19
0	1	0	2	1	1	0	14
0	1	1	3	1	1	1	15
0	1	-2	8	1	1	-2	16
0	1	-1	9	1	1	-1	17
0	2	0	4				
0	2	1	5				
0	2	-2	6				
0	2	-1	7				

6.53 LayoutElementProductionParams

`LayoutElementProductionParams` is needed for `LayoutElementProduction`. This resource contains detailed information about the type of `RunList` to be produced.

Resource Properties

Intent Pairing: [VariableIntent](#)

Input of Processes: [LayoutElementProduction](#)

Table 6.114: [LayoutElementProductionParams](#) Resource

NAME	DATA TYPE	DESCRIPTION
ContentRefs ?	IDREFS	@ ContentRefs SHALL reference Content that provide metadata related to the output RunList . Note: The referenced Content elements MAY contain OCGControl elements that select the desired optional content from versioned PDF files.
ShapeDefRef ?	IDREF	@ ShapeDefRef SHALL reference a ShapeDef that represents the shape of the RunList to be produced.
FileSpec (DataList) ?	element	References a data list containing record information for variable data production. The format of the referenced data is implementation specific.

Example 6.10: [LayoutElementProductionParams](#): Page Shape

The following example requests four pages with a trim size of A4 and a 5mm bleed.

Note: 14.17pts = 5mm.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0"
  JobID="LayoutElementProduction" JobPartID="PageSize" Types="LayoutElementProduction">
  <ResourceSet Name="LayoutElementProductionParams" Usage="Input">
    <Resource>
      <LayoutElementProductionParams ContentRefs="Content_000005.1"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Content">
    <Resource ID="Content_000005.1">
      <Content ContentType="Page" SourceClipBox="0 0 651.97 878.74" SourceTrimBox="14.17 14.17
595.28 850.39"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="RunList" Usage="Output">
    <Resource>
      <RunList NPage="4"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

Example 6.11: [LayoutElementProductionParams](#): Label Shape

The following example requests a label shape with an explicit triangular cut path.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0"
  JobID="LayoutElementProduction" JobPartID="ShapeDef" Types="LayoutElementProduction">
  <ResourceSet Name="LayoutElementProductionParams" Usage="Input">
    <Resource>
      <LayoutElementProductionParams ShapeDefRef="ShapeDef_000005.1"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="ShapeDef">
    <Resource ID="ShapeDef_000005.1">
      <ShapeDef>
        <Shape CutPath="10 0 1" DDESCutType="101" ShapeType="Path"/>
      </ShapeDef>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="RunList" Usage="Output">
    <Resource>
      <RunList NPage="1"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

Example 6.12: LayoutElementProductionParams: Box Shape

The following example requests a box that is described by an external CAD file.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0"
  JobID="LayoutElementProduction" JobPartID="ShapeDef" Types="LayoutElementProduction">
  <ResourceSet Name="LayoutElementProductionParams" Usage="Input">
    <Resource>
      <LayoutElementProductionParams ShapeDefRef="ShapeDef_000005.1"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="ShapeDef">
    <Resource ID="ShapeDef_000005.1">
      <ShapeDef>
        <FileSpec URL="file://myserver/myshare/olive.dd3"/>
      </ShapeDef>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="RunList" Usage="Output">
    <Resource>
      <RunList NPage="1"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

6.54 LayoutShift

LayoutShift defines the parameters for separation dependent paper stretch compensation.

Resource Properties

Input of Processes: **LayoutShifting**

Table 6.115: LayoutShift Resource

NAME	DATA TYPE	DESCRIPTION
ShiftPoint +	element	Description of separation dependent transformations for a given point on the Layout .

6.54.1 ShiftPoint

Table 6.116: 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 6.13: LayoutShift

Example of modifying the absolute positions of the "Black" separation with *ShiftPoint/@Position*.

```
<XJDF JobID="n_001007" JobPartID="n_000002" Types="Product" xmlns="http://www.CIP4.org/JDFSche-
ma_2_0">
  <ResourceSet Name="Layout" Usage="Input">
    <Resource>
      <Part SheetName="Sheet1" Side="Front"/>
      <Layout>
        <PlacedObject CTM="1 0 0 1 0 0" Ord="0">
          <ContentObject/>
        </PlacedObject>
        <PlacedObject CTM="1 0 0 1 700 900" Ord="0">
          <MarkObject/>
        </PlacedObject>
        <PlacedObject CTM="1 0 0 1 720 0" Ord="1">
          <ContentObject/>
        </PlacedObject>
        <PlacedObject CTM="1 0 0 1 0 1000" Ord="2">
          <ContentObject/>
        </PlacedObject>
        <PlacedObject CTM="1 0 0 1 720 1000" Ord="3">
          <ContentObject/>
        </PlacedObject>
      </Layout>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="LayoutShift" Usage="Input">
    <Resource>
      <Part Separation="Black" Side="Front"/>
      <LayoutShift>
        <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>
    </Resource>
  </ResourceSet>
</XJDF>
```

6.55 LooseBindingParams

LooseBindingParams describes the details of the *LooseBinding* process.

Resource Properties

Intent Pairing: *BindingIntent*

Input of Processes: *LooseBinding*

Table 6.117: *LooseBindingParams* Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>BindingType</i>	enumeration	Type of binding that is performed. Allowed values are: <i>ChannelBinding</i> – Metal clamps are used to bind sheets. <i>CoilBinding</i> – Metal wire, plastic coated wire or pure plastic wire 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.

Table 6.117: LooseBindingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>CoverMaterial</i> ?	NMTOKEN	@ <i>CoverMaterial</i> describes the binder materials used. Values include: <i>Cardboard</i> – Cardboard with no covering. <i>ClothCovered</i> – Cardboard with cloth covering. <i>Plastic</i> – Binder cover fabricated from solid plastic material (e.g., PVC). <i>PlasticCovered</i> – Cardboard with colored plastic covering.
<i>ChannelBindingDetails</i> ?	element	<i>ChannelBindingDetails</i> specifies additional details for channel binding. <i>ChannelBindingDetails</i> SHALL NOT be specified unless @ <i>BindingType</i> ="ChannelBinding".
<i>CoilBindingDetails</i> ?	element	<i>CoilBindingDetails</i> specifies additional details for coil binding. <i>CoilBindingDetails</i> SHALL NOT be specified unless @ <i>BindingType</i> ="CoilBinding".
<i>CombBindingDetails</i> ?	element	<i>CombBindingDetails</i> specifies additional details for comb binding. <i>CombBindingDetails</i> SHALL NOT be specified unless @ <i>BindingType</i> ="CombBinding".
<i>HolePattern</i> *	element	Details of the holes for binding. Values for <i>HolePattern</i> / <i>@Pattern</i> SHALL be taken from ▶ Appendix G Hole Pattern Catalog and SHALL follow the conventions specified in ▶ Appendix G.1 Naming Scheme.
<i>RingBindingDetails</i> ?	element	<i>RingBindingDetails</i> specifies additional details for ring binding. <i>RingBindingDetails</i> SHALL NOT be specified unless @ <i>BindingType</i> ="RingBinding".
<i>StripBindingDetails</i> ?	element	<i>StripBindingDetails</i> specifies additional details for strip binding. <i>StripBindingDetails</i> SHALL NOT be specified unless @ <i>BindingType</i> ="StripBinding".

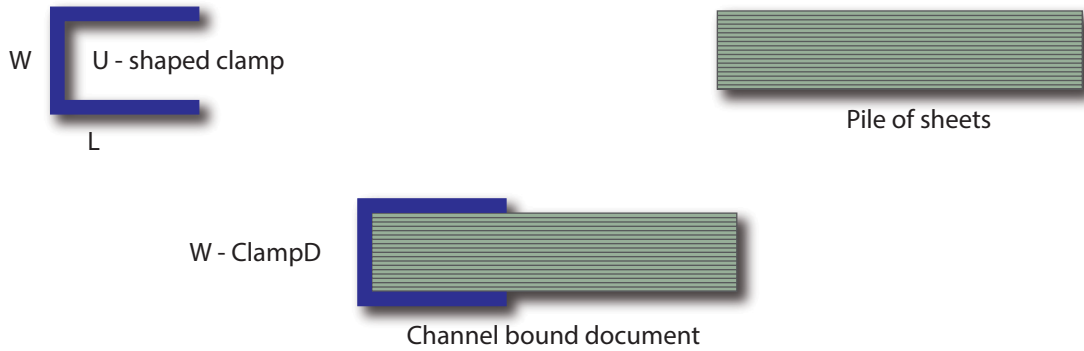
6.55.1 ChannelBindingDetails

▶ Figure 6-28: Parameters used for channel binding depicts the channel binding with a clamp. The symbols W, L and ClampD of ▶ Figure 6-28: Parameters used for channel binding are described by the attributes @*ClampD* and @*ClampSize* of the table below.

Table 6.118: ChannelBindingDetails Element

NAME	DATA TYPE	DESCRIPTION
<i>ClampD</i> ?	float	The distance of the clamp that was “pressed away”, see ▶ Figure 6-28: Parameters used for channel binding.
<i>ClampSize</i> ?	shape	The shape size of the clamp. The first number of the shape data type corresponds to the clamp width W (see ▶ Figure 6-28: Parameters used for channel binding) 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 6-28: Parameters used for channel binding). The third corresponds to the spine length (not visible in ▶ Figure 6-28: Parameters used for channel binding). The spine length is perpendicular on the paper plane.
<i>Cover</i> ?	boolean	If "true" the clamp is inside of a preassembled cover.

Figure 6-28: Parameters used for channel binding



6.55.2 CoilBindingDetails

Table 6.119: CoilBindingDetails Element

NAME	DATA TYPE	DESCRIPTION
<i>CoilShape</i> ?	NMTOKEN	The shape of the wire coil binding. Values include those from: ▶ Comb and Coil Shapes.
<i>Diameter</i> ?	float	Specifies the diameter of the comb to be produced. The diameter is determined by the height of the block of sheets to be bound.

6.55.3 CombBindingDetails

Table 6.120: CombBindingDetails Element

NAME	DATA TYPE	DESCRIPTION
<i>CombShape</i> ?	NMTOKEN	The shape of the wire comb binding. Allowed value includes those from: ▶ Comb and Coil Shapes.
<i>Diameter</i> ?	float	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>FlipBackCover</i> ?	boolean	The spine is typically hidden between the last page of the Component and the back cover. <i>@FlipBackCover="true"</i> specifies that the cover SHALL be flipped after the wire was “closed”, otherwise the cover SHALL be kept open. The latter makes sense if further processing is needed before closing the book, e.g. inserting a CD.

6.55.4 RingBindingDetails

Table 6.121: RingBindingDetails Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Diameter</i> ?	float	Specifies the diameter of the rings, in points.
<i>RingMechanic</i> ?	boolean	If "true", a hand lever is available for opening.
<i>RingShape</i> ?	NMTOKEN	<i>@RingShape</i> specifies the shape of the ring binder rings. Values include: D-shape Oval Round SlantD

Table 6.121: RingBindingDetails Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>RivetsExposed</i> ?	boolean	@ <i>RivetsExposed</i> describes the mounting of the ring mechanism in the 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.
<i>SpineWidth</i> ?	float	The spine width is determined by the final height of the block of sheets to be bound.
<i>ViewBinder</i> ?	NMTOKEN	@ <i>ViewBinder</i> specifies the details of the clear vinyl outer-wrap types on top of a colored base wrap: Values include: <i>Embedded</i> – Printed material is embedded by sealing between the colored and clear vinyl layers during the binder manufacturing. <i>Pocket</i> – Binder is designed so that printed material can be inserted between the colored and clear vinyl layers after the binder is manufactured.

6.55.5 StripBindingDetails

Table 6.122: StripBindingDetails Element

NAME	DATA TYPE	DESCRIPTION
<i>Length</i> ?	float	The length of the pin is determined by the height of the pile of sheets to be bound.

6.56 ManualLaborParams

ManualLaborParams 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

Input of Processes: **ManualLabor**

Table 6.123: ManualLaborParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>LaborType</i>	NMTOKEN	Type of manual labor that is performed. Values include: <i>CarvePotato</i> – Carve a potato for potato printing. <i>CreateCoatingForm</i> – Create a form to apply coatings during or after printing. <i>EditArt</i> – Unspecific art editing (for work on specific files LayoutElementProduction SHALL be used). <i>EditMarks</i> – Marks editing. <i>EditTraps</i> – Traps editing. <i>ManageJob</i> – General work on the job. <i>PhoneCallToCustomer</i> – Phone calls to ask/inform the customer. <i>SeparateBlanks</i> – Manual separation of blanks from a sheet after die cutting.

6.57 Media

Media represents the properties of a raw, unexposed printable surface such as a paper sheet, film or plate.

Note: When using a substrate such as paper for printing, *Media* represents the properties of that substrate, whereas *Component* represents the physical substrate itself.

Properties

Resource referenced by: **ConvertingConfig, DieLayout, Media/MediaLayers**

Intent Pairing: **MediaIntent, HoleMakingIntent,**

Input of Processes: **BoxPacking, Bundling, Embossing, ImageSetting, Laminating**

Table 6.124: Media Resource (Sheet 1 of 4)

NAME	DATA TYPE	DESCRIPTION
<i>BackBrightness</i> ?	float	Equivalent to <i>@Brightness</i> (see below), but applied to the back surface of the <i>Media</i> . If not specified, the value of <i>@Brightness</i> SHALL be applied to the front and back surfaces of the <i>Media</i> .
<i>BackCoating</i> ?	enumeration	Identical to <i>@Coating</i> (see below), but applied to the back surface of the media. If not specified, the value of <i>@Coating</i> SHALL be applied to the front and back surfaces of the <i>Media</i> . Allowed value is from: ▶ Coating.
<i>BackCoatingDetail</i> ?	NMTOKEN	Identical to <i>@CoatingDetail</i> (see below), but applied to the back surface of the media. If not specified, the value of <i>@CoatingDetail</i> SHALL be applied to the front and back surfaces of the <i>Media</i> . Values include: Cast
<i>BackGlossValue</i> ?	float	Identical to <i>@GlossValue</i> (see below), but applied to the back surface of the media. If not specified, the value of <i>@GlossValue</i> SHALL be applied to the front and back surfaces of the <i>Media</i> .
<i>BackISOPaperSubstrate</i> ?	enumeration	Identical to <i>@ISOPaperSubstrate</i> (see below), but applied to the back surface of paper material defined in accordance with the print substrate set forth in ▶ [ISO12647-2:2013]. If not specified, the value of <i>@ISOPaperSubstrate</i> SHALL be applied to the front and back surfaces of the <i>Media</i> . Allowed value is from: ▶ ISOPaperSubstrate.
<i>Brightness</i> ?	float	Reflectance percentage of diffuse blue reflectance as defined by ▶ [ISO2470-1:2016]. The reflectance is reported per ▶ [ISO2470-1:2016] as the diffuse blue reflectance factor of the media in percent to the nearest 0.5% reflectance factor. See also <i>@BackBrightness</i> .
<i>CIEtint</i> ?	float	Average CIE tint value. Average CIE tint is calculated according to equations given in ▶ [TAPPI T560].
<i>CIEWhiteness</i> ?	float	Average CIE whiteness value. Average CIE whiteness is calculated according to equations given in ▶ [TAPPI T560].
<i>Coating</i> ?	enumeration	The pre-process coating that has been applied to the media. Allowed value is from: ▶ Coating.
<i>CoatingDetail</i> ?	NMTOKEN	Describes additional details of the coating that has been applied to the media and possibly the technology used to apply the coating. Values include: Cast
<i>CoreWeight</i> ?	float	Weight of the core of a roll, in grams [g].
<i>Dimension</i> ?	XYPair	The X and Y dimensions of the <i>Media</i> , measured in points. <i>@Dimension</i> specifies the outer bounding box of the <i>Media</i> . 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 "Roll" 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 "0". If either or both X or Y="0" (i.e., unknown), the default orientation is assumed to be portrait (i.e., Y > X). Values include those from: ▶ Appendix D Media Size.
<i>Flute</i> ?	NMTOKEN	Single, capital letter that specifies the flute type of corrugated media. See ▶ Section 6.57.2.2 Corrugated Media. Values include those from: ▶ Flute Types.
<i>FluteDirection</i> ?	enumeration	Direction of the flute of corrugated media in the coordinate system of the <i>Media</i> . See ▶ Section 6.57.2.2 Corrugated Media. Allowed value is from: ▶ MediaDirection. A value of "SameDirection" SHALL NOT be specified.

Table 6.124: Media Resource (Sheet 2 of 4)

NAME	DATA TYPE	DESCRIPTION
<i>GlossValue</i> ?	float	Gloss of the media in gloss units as defined by ▶ [ISO8254-1:2009]. Refer also to ▶ [TAPPI T480] for examples of gloss calculation.
<i>GrainDirection</i> ?	enumeration	Direction of the grain in the coordinate system of the <i>Media</i> . A value of "SameDirection" SHALL NOT be specified. Allowed value is from: ▶ MediaDirection.
<i>ImagableSide</i> ?	enumeration	Side of the chosen medium that can be marked. Allowed values are: Front Back Both Neither
<i>InnerCoreDiameter</i> ?	float	Specifies the inner diameter of the core of a roll, in points. See also @OuterCoreDiameter and @RollDiameter.
<i>InsideLoss</i> ?	float	The inside loss of corrugated board material in microns [µm]. See ▶ Section 6.57.2.1 Inside Loss and Outside Gain. Note: @InsideLoss + @OutsideGain NEED NOT be exactly equal to thickness.
<i>ISOPaperSubstrate</i> ?	enumeration	@ISOPaperSubstrate SHALL specify the type of paper material defined in ▶ [ISO12647-2:2013]. Allowed value is from: ▶ ISOPaperSubstrate.
<i>LabColorValue</i> ?	LabColor	@LabColorValue is the CIELAB color value of the media, computed as specified in ▶ [TAPPI T527].
<i>MediaColorName</i> ?	enumeration	A name for the color. If more specific, specialized or site-defined media color names are needed, use @MediaColorNameDetails. Allowed value is from: ▶ NamedColor.
<i>MediaColorNameDetails</i> ?	string	A more specific, specialized or site-defined name for the media color. If @MediaColorNameDetails is supplied, @MediaColorName SHOULD also be supplied. @MediaColorNameDetails SHOULD be used to specify specialized media properties such as holographic or transparent foils.
<i>MediaQuality</i> ?	string	Named quality description of the media. Media with the same properties except for @Dimension SHOULD have the same value of @MediaQuality. For folding carton quality, multiple named quality description systems are in use (e.g., GC1, SBB, etc.). For an overview, see ▶ [Pro Carton].
<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 @MediaTypeDetails is "PreCutTabs", a @MediaSetCount of "5" would indicate that each set includes five tab sheets.
<i>MediaType</i>	enumeration	Describes the general type of the <i>Media</i> . Allowed value is from: ▶ MediaType.
<i>MediaTypeDetails</i> ?	NMTOKEN	Additional details of the chosen medium. Values include those from: ▶ MediaType Details.
<i>MediaUnit</i> ?	enumeration	Describes the format of the media as it is delivered to the device. Allowed values are: Continuous – Continuously connected sheets that can be fan folded. Roll – Continuous web on a reel. Sheet – Individual cut sheets.
<i>Opacity</i> ?	enumeration	The opacity of the media. See @OpacityLevel to specify the degree of opacity for any of these values. Allowed value is from: ▶ Opacity.
<i>OpacityLevel</i> ?	float	Normalized TAPPI opacity (Cn), as defined and computed in ▶ [ISO2471:2008]. Refer also to ▶ [TAPPI T519] for calculation examples.

Table 6.124: Media Resource (Sheet 3 of 4)

NAME	DATA TYPE	DESCRIPTION
<i>OuterCoreDiameter</i> ?	float	Specifies the outer diameter of the core of a roll, in points. See also <i>@InnerCoreDiameter</i> and <i>@RollDiameter</i> .
<i>OutsideGain</i> ?	float	The outside gain of corrugated board material in microns [µm]. See ▶ Section 6.57.2.1 Inside Loss and Outside Gain. Note: <i>@InsideLoss</i> + <i>@OutsideGain</i> NEED NOT be exactly equal to thickness.
<i>PlateTechnology</i> ?	enumeration	Exposure technology of the plates. Allowed values are: <i>FlexoAnalogSolvent</i> <i>FlexoAnalogThermal</i> <i>FlexoDigitalSolvent</i> <i>FlexoDigitalThermal</i> <i>FlexoDirectEngraving</i> <i>InkJet</i> – Exposure with inkjet technology. Note that <i>@PrintingTechnology="InkJet"</i> specifies paper or transparency media, not plates. <i>Thermal</i> – Thermal exposure. <i>UV</i> – Ultraviolet exposure. <i>Visible</i> – Visible light exposure.
<i>Polarity</i> ?	enumeration	Polarity of the chosen medium. Allowed value is from: ▶ <i>Polarity</i> .
<i>PrintingTechnology</i> ?	NMTOKEN	Describes the printing technology that the media or coatings on the media are intended for or optimized for. Values include those from: ▶ <i>Printing Technologies</i> .
<i>RecycledPercentage</i> ?	float	The percentage, between 0 and 100, of recycled material that the media SHALL contain.
<i>ReliefThickness</i> ?	float	The thickness of the relief, measured in microns [µm]. The floor thickness can be calculated as (<i>@Thickness</i> - <i>@ReliefThickness</i>). See ▶ Figure 6-30: Relief and floor thickness for a flexo plate or flexo sleeve.
<i>RollDiameter</i> ?	float	Specifies the diameter of a roll, in points. See also <i>@InnerCoreDiameter</i> and <i>@OuterCoreDiameter</i> .
<i>ShrinkIndex</i> ?	XYPair	Specifies the ratio of the media linear dimension after shrinking to that prior to shrinking. The X value specifies the 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	The type of interlock (or notch) to use for a flexo sleeve.
<i>StockType</i> ?	NMTOKEN	<i>@StockType</i> defines the base size when calculating North American or Japanese paper weights. See ▶ Section C Media Weight for details including pre-defined values.
<i>Texture</i> ?	NMTOKEN	The texture of paper media. Values include those from: ▶ <i>Texture</i> .
<i>Thickness</i> ?	float	The thickness of the chosen medium, measured in microns [µm]. Note: Thickness is often referred to as caliper.
<i>Weight</i> ?	float	Weight of the chosen medium, measured in grams per square meter [g/m ²]. See ▶ Appendix C Media Weight for details on converting anachronistic paper weights to g/m ² .
<i>Certification</i> *	element	<i>Certification</i> SHALL specify a paper certification level that the paper fulfills.
<i>HolePattern</i> *	element	List of holes in the <i>Media</i> .
<i>IdentificationField</i> *	element	<i>IdentificationField</i> associates bar codes or labels with this <i>Media</i> .

Table 6.124: Media Resource (Sheet 4 of 4)

NAME	DATA TYPE	DESCRIPTION
MediaLayers ?	element	MediaLayers describes the layer structure of media such as corrugated or self adhesive materials.
TabDimensions ?	element	Specifies the dimensions of the tabs when <code>@MediaTypeDetails="TabStock"</code> , <code>"PreCutTabs"</code> or <code>"FullCutTabs"</code> . Note: See BindingIntent/Tabs (▶ Table 4.20 Tabs Element) (rather than MediaIntent) for how tabbed media is specified in product intent.

Figure 6-29: Paper roll with some roll-specific information

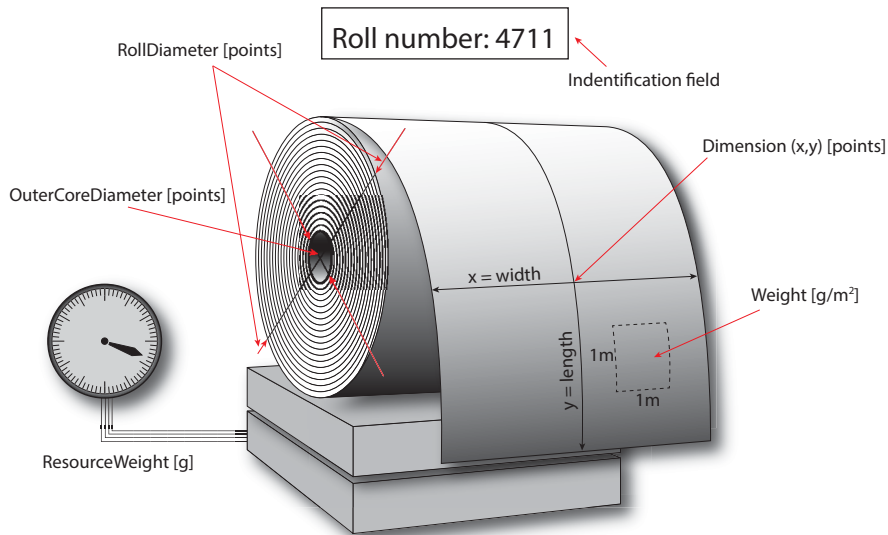
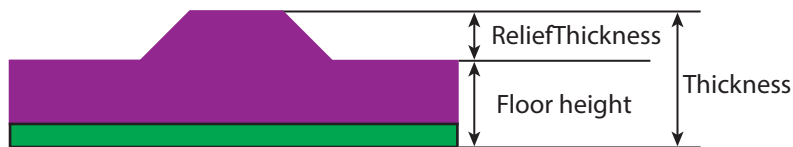


Figure 6-30: Relief and floor thickness for a flexo plate or flexo sleeve



6.57.1 TabDimensions

Specifies the size and placement of tabs in a bank and in a set of tab stock.

Table 6.125: TabDimensions Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
TabEdge ?	enumeration	Indicates which edge of the media has tabs. Sets the coordinate system for <code>@TabOffset</code> , <code>@TabExtensionDistance</code> , and <code>@TabWidth</code> . Allowed value is : ▶ Edge.
TabExtensionDistance ?	float	The positive distance in points that the tab extends beyond the body of the other media. Note: Same as BindingIntent/Tabs/@TabExtensionDistance . Note: This value is always included in the value of the overall extent of the Media defined by Media/@Dimension . See ▶ Figure 6-31: Diagram of a single bank of tabs.

Table 6.125: TabDimensions Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>TabOffset</i> ?	float	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 @TabEdge. This distance is the same on both ends of the bank of tabs. See ▶ Figure 6-31: Diagram of a single bank of tabs.
<i>TabSetCollationOrder</i> ?	enumeration	Collation order of media provided in sets. Applicable to sets of pre-cut tabs. See ▶ Figure 6-31: Diagram of a single bank of tabs. Allowed values is from: ▶ Table 6.126 TabSetCollationOrder Attribute Values.
<i>TabsPerBank</i> ?	integer	Specifies the number of equal-sized tabs in a single bank if all positions were filled. Note: Banks can have tabs only in some of the possible positions. Note: Same as <i>BindingIntent/Tabs/@TabsPerBank</i> . <i>Media/@MediaSetCount</i> specifies the number of tabs per set. A set can consist of one or more banks. If <i>Media/@MediaSetCount</i> is not an even multiple of @TabsPerBank, the last bank in each set is partially filled.
<i>TabWidth</i> ?	float	The width along the @TabEdge 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 6-31: Diagram of a single bank of tabs.

Table 6.126: TabSetCollationOrder Attribute Values

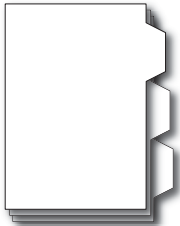
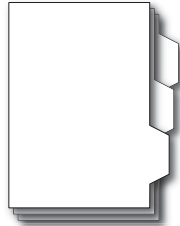
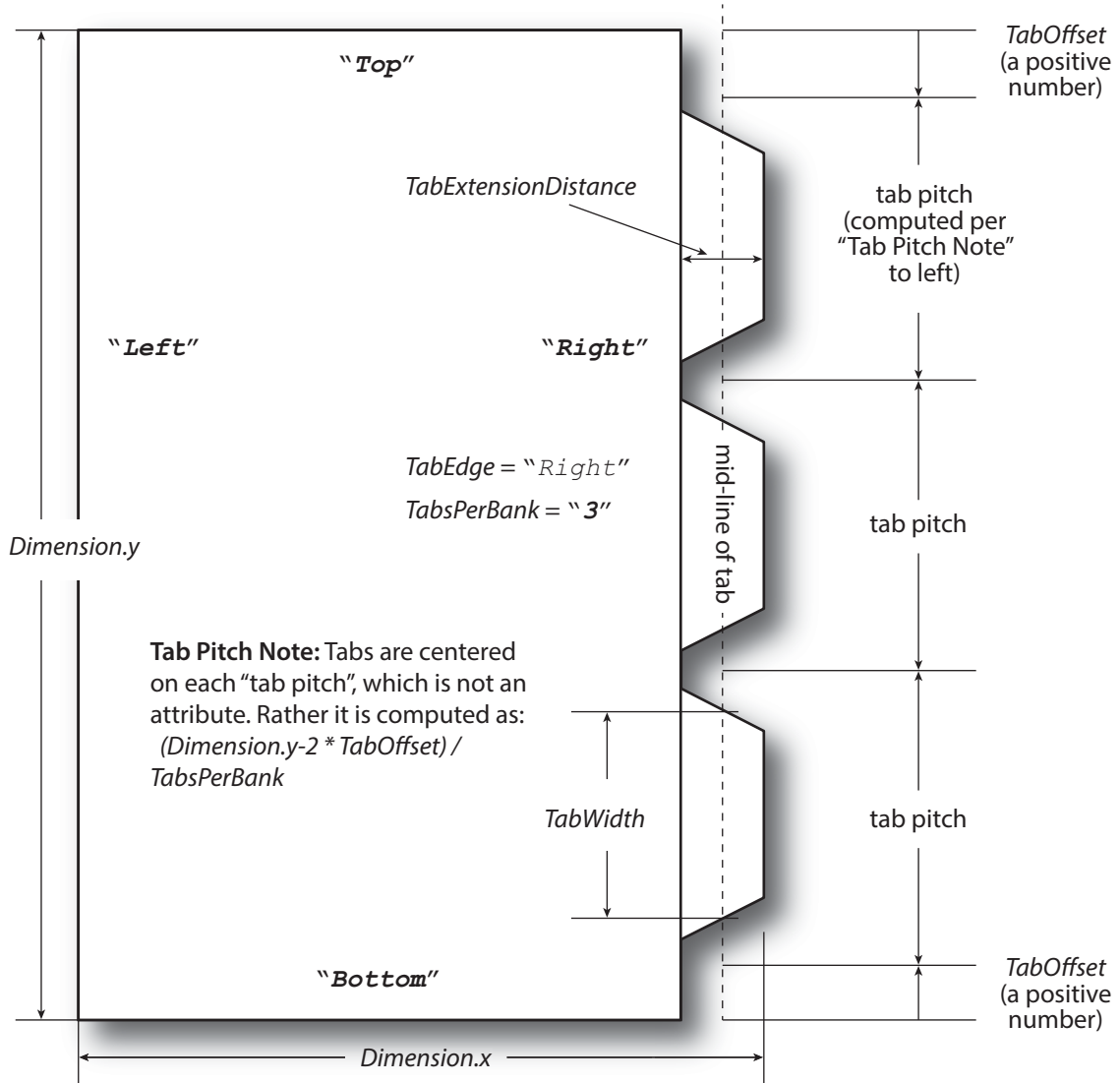
VALUE	DESCRIPTION	EXAMPLE
Forward	The first tab in reader order SHALL be placed towards the top of the stack.	
Reverse	The first tab in reader order SHALL be placed towards the bottom of the stack.	

Figure 6-31: Diagram of a single bank of tabs



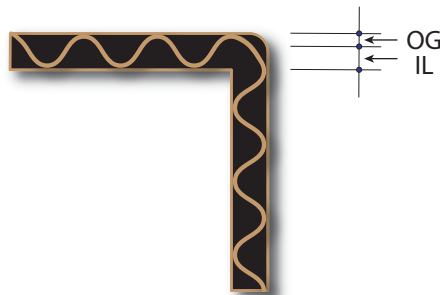
6.57.2 More about Media

6.57.2.1 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 6-32: Inside Loss, Outside Gain



6.57.2.2 Corrugated Media

Corrugated material consists of multiple sheets of paper (called liners) with fluted material in between. For background information on corrugated media, see ▶ [Corrugated Packaging]. 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 ▶ [Corrugated Packaging].

Example 6.14: Media: Corrugated

The following example describes single wall corrugated media with two liners and a "B" type flute.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="Converting"
  JobPartID="Corrugated" Types="ConventionalPrinting">
  <ResourceSet Name="Media" Usage="Input">
    <Resource>
      <Media Dimension="2834.64566929 1984.2519685" InsideLoss="1000"
        MediaType="CorrugatedBoard" MediaTypeDetails="SingleWall"
        OutsideGain="1380" Thickness="2382">
        <MediaLayers>
          <Media MediaType="Paper" Weight="190"/>
          <Media Flute="B" FluteDirection="XDirection" MediaType="Paper"
            MediaTypeDetails="Flute" Weight="180"/>
          <Media MediaType="Paper" Weight="180"/>
        </MediaLayers>
      </Media>
    </Resource>
  </ResourceSet>
</XJDF>
```

6.57.2.3 Self adhesive Media

Self adhesive media is described as a *MediaLayers* element with nested *Media* and *Glue* elements.

Example 6.15: Media: Self Adhesive

The following example describes labels with removable glue on a 60 gram base.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="Converting"
  JobPartID="Corrugated" Types="ConventionalPrinting">
  <ResourceSet Name="Media" Usage="Input">
    <Resource>
      <Media Dimension="1190.5511811 0" MediaType="SelfAdhesive"
        MediaUnit="Roll" Thickness="900">
        <MediaLayers>
          <Media MediaType="Paper" Weight="90"/>
          <Glue AreaGlue="true" GlueType="Removable"/>
          <Media MediaType="Paper" Weight="60"/>
        </MediaLayers>
      </Media>
    </Resource>
  </ResourceSet>
</XJDF>
```

6.57.2.4 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 photo-polymer.

Example 6.16: Media: Flexo Plate

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="Flexo"
  JobPartID="Plate" Types="ConventionalPrinting">
  <ResourceSet Name="Media" ProcessUsage="Plate" Usage="Input">
    <Resource Brand="FlexoBrand">
      <Part Separation="Black"/>
      <Media Dimension="2551.18110236 3401.57480315" MediaType="Plate"
        PlateTechnology="FlexoDigitalThermal" ReliefThickness="500" Thickness="1143">
        <MediaLayers>
          <Media MediaType="Plate" MediaTypeDetails="FlexoPhotoPolymer" Thickness="966"/>
          <Media MediaType="Plate" MediaTypeDetails="FlexoBase" Thickness="177"/>
        </MediaLayers>
      </Media>
    </Resource>
  </ResourceSet>
</XJDF>
```

6.57.2.5 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 the printing dimensions (no distortion).

Example 6.17: Media: Flexo Sleeve

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="Flexo"
  JobPartID="Sleeve" Types="ConventionalPrinting">
  <ResourceSet Name="Media" ProcessUsage="Plate" Usage="Input">
    <Resource Brand="FlexoBrand">
      <Media Dimension="1417.32283465 708.66141732" MediaType="Sleeve"
        PlateTechnology="FlexoDigitalSolvent" ReliefThickness="500" Thickness="2810">
        <MediaLayers>
          <Media MediaType="Sleeve" MediaTypeDetails="FlexoPhotoPolymer" Thickness="1570"/>
          <Media MediaType="Sleeve" MediaTypeDetails="FlexoBase" Thickness="1249"/>
        </MediaLayers>
      </Media>
    </Resource>
  </ResourceSet>
</XJDF>
```

6.58 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 SHOULD not be used to describe resources that are already more specifically defined in **XJDF** such as **Ink**, **Media**, **Pallet**, **RegisterRibbon** 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[@Type="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

Input of Processes: **Any Process**

Table 6.127: MiscConsumable Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Color</i> ?	enumeration	@Color specifies the machine readable color of the consumable. Allowed value is from: ▶ NamedColor.
<i>ColorDetails</i> ?	string	@ColorDetails specifies additional details of the color of the consumable that MAY be site specific and MAY be human readable. @Color SHOULD be specified if @ColorDetails is supplied.

Table 6.127: MiscConsumable Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
Type	NMTOKEN	Identifies the type of MiscConsumable (machine-readable). A human-readable description of the consumable SHOULD also be supplied in Resource/@DescriptiveName . Additional machine readable details MAY be provided in @TypeDetails . Values include those from: ▶ Table 6.128 Type Attribute Values.
TypeDetails ?	NMTOKEN	Additional details of the consumable such as material.
IdentificationField *	element	IdentificationField associates bar codes or labels with this MiscConsumable .

Table 6.128: Type Attribute Values

VALUE	DESCRIPTION
BackReinforcement	Strip of material that is used to reinforce the spines of a hardcover book.
BlisterPack	Air filled filler material, e.g. as used for BoxPacking .
ChannelBinder	Metal clamp used for channel binding. See LooseBinding for details.
Coil	Metal or plastic wire coil used for coil binding. See LooseBinding for details.
Comb	Metal or plastic comb used for comb binding. See LooseBinding for details.
Cover	Additional cover used for loose binding. See LooseBinding for details.
Developer	Chemicals used in filmsetters and platesetters.
DigitalMedia	Digital media represents removable digital media such as thumb drives or removable disks. Examples values for @TypeDetails include: CD - Compact disk. DVD SDCard - A memory card, e.g. from a digital camera. USBDrive - A USB attached disk drive.
Electricity	Electrical energy. Typically monitored for CO2 tracking. Measured in kWh.
FuserOil	Silicon oil.
Gas	Natural gas. Typically monitored for CO2 tracking. Measured in m ³ .
Glue	Glue is used in many postpress processes.
Grommet	Specifies an eyelet-like shape placed in a hole.
Hardener	Glue hardener used in two component adhesives.
Headband	Head band for a hardcover book.
MountingTape	Mounting tape used for a sleeve in Flexo printing.
Paper	Unprinted paper, e.g. as used for filling material in BoxPacking .
PaperBand	Paper band, e.g. as used for Wrapping .
PaperWrap	Paper wrapper, e.g. as used for Wrapping .
PlasticBand	Plastic band, e.g. as used for Wrapping .
RegisterRibbon	Register ribbons in books. See BlockPreparation for details.
RingBinder	Ring binder used for ring binding. See LooseBinding for details.
RubberBand	Rubber band, e.g. as used for Wrapping .

Table 6.128: Type Attribute Values

VALUE	DESCRIPTION
ShrinkWrap	Shrink wrapping material, e.g. as used for Wrapping .
Staples	Prefabricated staples used for Stitching . Use Wire when the staples are not prefabricated.
Strap	Straps are used in a Strapping process.
StripBinder	Strip binder used for strip binding. See LooseBinding for details.
Styrofoam	Styrofoam, e.g. as used for BoxPacking .
Tape	Adhesive tape, e.g. as used for SpineTaping .
Thread	Thread used for ThreadSealing or ThreadSewing .
WasteContainer	Waste container, e.g. for used ink or toner.
Wire	Bulk wire used for forming staples or other bindings.

6.59 NodeInfo

The **NodeInfo** resource contains information about planned scheduling. It allows MIS to plan, schedule and invoice jobs or job parts.

Resource Properties

Input of Processes: **Any Process**

Table 6.129: NodeInfo Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>CleanupDuration</i> ?	duration	Estimated duration of the clean-up phase of the process.
<i>DueLevel</i> ?	enumeration	Description of the severity of a missed deadline. Allowed values are: JobCancelled – The job is cancelled if the deadline is missed. Penalty – Missing the deadline incurs a penalty. Trivial – Missing the deadline has minor or no consequences.
<i>End</i> ?	dateTime	Date and time at which the process is scheduled to end.
<i>FirstEnd</i> ?	dateTime	Earliest date and time at which the process SHALL end.
<i>FirstStart</i> ?	dateTime	Earliest date and time at which the process SHALL begin.
<i>JobPriority</i> ?	integer	The scheduling priority for the node where 100 is the highest and 0 is the lowest. 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 XJMF (<i>QueueSubmissionParams</i> / <i>@Priority</i> or <i>ModifyQueueEntryParams</i> / <i>@Priority</i>) SHALL take precedence over NodeInfo / <i>@JobPriority</i> .
<i>LastEnd</i> ?	dateTime	Latest date and time at which the process SHALL end. This is the deadline to which <i>@DueLevel</i> refers.
<i>LastStart</i> ?	dateTime	Latest date and time at which the process SHALL begin.
<i>NaturalLang</i> ?	language	Language selected for human readable communication. If not specified, the operating system's language SHOULD be used.
<i>PersonalID</i> ?	NMTOKEN	<i>Resource</i> / <i>@ExternalID</i> of the Contact that represents the employee that is responsible for processing this XJDF .
<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.

Table 6.129: NodeInfo Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Status</i> ?	enumeration	Identifies the status of an individual part of the XJDF . Allowed value is from: ▶ Status.
<i>StatusDetails</i> ?	NMTOKEN	Machine readable description of the status that provides details beyond the enumerative values given by <i>@Status</i> . Values include those from: ▶ Status Details.
<i>TotalDuration</i> ?	duration	Estimated total duration of the process, including setup and cleanup.
<i>GangSource</i> *	element	If present, each <i>GangSource</i> SHALL represent the source jobs that are being processed as a gang job.
<i>MISDetails</i> ?	element	Definition how the costs for the execution of this node SHALL be charged.

6.60 Pallet

A *Pallet* represents the pallet used in packing goods.

Resource Properties

Input of Processes: **Palletizing**

Table 6.130: Pallet Resource

NAME	DATA TYPE	DESCRIPTION
<i>PalletType</i>	NMTOKEN	Type of pallet used. Values include those from: ▶ Pallet Types.
<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> .
<i>IdentificationField</i> *	element	<i>IdentificationField</i> associates bar codes or labels with this <i>Pallet</i> .

6.61 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

Input of Processes: **Palletizing**

Table 6.131: 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> ?	float	Maximum height of a loaded pallet in points.
<i>MaxWeight</i> ?	float	Maximum weight of a loaded pallet in grams.
<i>Overhang</i> ?	XYPair	Overhang in x and y direction on each side.
<i>OverhangOffset</i> ?	XYPair	Overhang offset if overhang is not centered.
<i>Pattern</i> ?	NMTOKEN	Name of the palletizing pattern. Used to store a predefined pattern that defines the layers and positioning of individual component on the pallet.

6.62 PDLCreationParams

PDLCreationParams describes the details of generating the supported output PDL types used in the **PDLCreation** process.

Resource Properties

Input of Processes: **PDLCreation**

Table 6.132: PDLCreationParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>MimeType</i>	string	This resource identifies the MIME type associated with this output file format. For example " application/pdf ". IANA maintains a registry of MIME types, see ▶ [IANA-mt].
<i>FontParams</i> ?	element	FontParams describes how fonts SHALL be handled when creating PDL.
<i>PDFCreationDetails</i> ?	element	PDF specific element for the output. It SHALL NOT be specified unless @MimeType="application/pdf" .
<i>PSCreationDetails</i> ?	element	Postscript specific element for the output. It SHALL NOT be specified unless @MimeType="application/postscript" .

6.62.1 AdvancedParams

Table 6.133: AdvancedParams Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>AllowPSXObject</i> s ?	boolean	If " false ", PostScript XObjects SHALL not be created.
<i>AllowTransparency</i> ?	boolean	If " false ", transparency SHALL not be present in the PDF.
<i>AutoPositionEPSInfo</i> ?	boolean	If " true ", the process SHALL automatically resize and center 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 SHALL be embedded in the PDF.
<i>EmitDSCWarnings</i> ?	boolean	If " true ", warning messages about questionable or incorrect DSC comments MAY 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) SHALL be ignored. If " false ", each parameter defined in the source document SHALL override that set in the XJDF .
<i>ParseDSCComment</i> or <i>DocInfo</i> ?	boolean	If " true ", the process SHALL parse the DSC comments in a PostScript source file and extract the document information. This information SHALL be recorded in the info dictionary of the PDF file.
<i>ParseDSCComments</i> ?	boolean	If " true ", the process SHALL parse 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 SHALL treat the DSC comments as pure PS comments and SHALL ignore them. (PostScript source only)
<i>PassThroughJPEGIm</i> ages ?	boolean	If " true ", JPEG images SHALL be passed through without recompressing them.
<i>PreserveCopyPage</i> ?	boolean	If " true ", the copypage operator of PostScript Level 2 SHALL be maintained. If " false ", the PostScript Level 3 definition of the copypage operator SHALL be 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)

Table 6.133: *AdvancedParams* Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>PreserveEPSInfo</i> ?	boolean	If "true", the EPS information in a PostScript source file SHALL be preserved and SHALL be stored in the resulting PDF file. (PostScript source only)
<i>PreserveHalftoneInfo</i> ?	boolean	If "true", the halftone screen information (frequency, angle and spot function) SHALL be passed into the PDF file. If "false", halftone information SHALL not be passed on.
<i>PreserveOPIComments</i> ?	boolean	If "true", Open Prepress Interface (OPI) low resolution images SHALL be encapsulated as a form and information for locating the high resolution images SHALL be preserved.
<i>PreserveOverprintSettings</i> ?	boolean	If "true", the value of the setoverprint operator SHALL be passed through to the PDF file. Otherwise, overprint SHALL be ignored.
<i>TransferFunctionInfo</i> ?	enumeration	Determines how transfer functions are handled. Allowed values are: Apply – 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. Preserve – Transfer functions are passed into the PDF file. Remove – Transfer functions are ignored. They are neither applied to the color values nor passed into the PDF file.
<i>UCRandBGInfo</i> ?	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) SHALL be passed into the PDF file. Allowed values are: Preserve – The arguments SHALL be passed into the PDF file. Remove – The arguments SHALL be ignored.
<i>UsePrologue</i> ?	boolean	If "true", the process SHALL append a PostScript prologue file before the 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)

6.62.2 FontParams

This element describes how fonts are handled when converting PostScript or other PDL files to PDF.

Table 6.134: *FontParams* Element

NAME	DATA TYPE	DESCRIPTION
<i>AlwaysEmbed</i> ?	NMTOKENS	<i>@AlwaysEmbed</i> specifies a list of one or more names of fonts that SHALL be embedded in the PDF. Each name SHALL be the PDL name of the font. Font names SHALL NOT occur in both the <i>@AlwaysEmbed</i> and <i>@NeverEmbed</i> lists.
<i>EmbedAllFonts</i> ?	boolean	If "true", specifies that all fonts, except those in the <i>@NeverEmbed</i> list, SHALL be embedded in the PDF.
<i>MaxSubsetPct</i> ?	integer	If the percentage of glyphs used from a font is below the value of <i>@MaxSubsetPct</i> , then a subset of the font SHALL be embedded in the PDF.
<i>NeverEmbed</i> ?	NMTOKENS	<i>@NeverEmbed</i> specifies a list of one or more names of fonts that SHALL NOT be embedded in the PDF and SHALL NOT be considered for subsetting. Each name SHALL be the PDL name of the font. Font names SHALL NOT occur in both the <i>@AlwaysEmbed</i> and <i>@NeverEmbed</i> lists.

6.62.3 PDFCreationDetails

This element contains the parameters that control the conversion of 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 parameters apply only to PostScript sources.

Table 6.135: PDFCreationDetails Element

NAME	DATA TYPE	DESCRIPTION
AllowJBIG2Globals ?	boolean	If "true", JBIG2 compressed images MAY share a single global dictionary in the resulting PDF file instead of a dictionary per image.
ASCII85EncodePages ?	boolean	If "true", binary streams (e.g., page content streams, sampled images, and embedded fonts) SHALL be ASCII85-encoded, resulting in a PDF file that is almost pure ASCII. If "false", they NEED NOT be encoded, resulting in a PDF file that can contain substantial amounts of binary data.
AutoRotatePages ?	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: All – Takes the predominant text orientation across all pages and rotates all pages the same way. None – Turns @AutoRotatePages off. PageByPage – Does the rotation on a page-by-page basis, rotating each page individually. Useful for documents that use both portrait and landscape orientations.
Binding ?	enumeration	Determines how the printed pages SHALL be bound. Allowed values are: Left – For left binding. Right – For right binding.
CompressPages ?	boolean	Enables compression of pages and other content streams like forms, patterns and Type 3 fonts. If "true", use Flate compression.
DefaultRenderingIntent ?	enumeration	Selects the rendering intent for the current job. See ▶ [PDF1.6] for more information on rendering intent. Allowed value is from: ▶ RenderingIntent.
DetectBlend ?	boolean	Enables or disables blend detection. If "true" and if @PDFVersion is 1.3 or higher, then blends will be converted to smooth shadings.
DoThumbnails ?	boolean	If "true", thumbnails are created.
InitialPageSize ?	XYPair	Defines the initial page dimensions, in points, that will be used to set Media-Box. 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 @InitialPageSize).
InitialResolution ?	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.
Optimize ?	boolean	If "true", the PDF file SHALL be optimized for size. See ▶ [PDF1.6] for more information on optimization.
OverPrintMode ?	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 ▶ [ColorPS].
PDFVersion ?	NMTOKEN	Specifies the version number of the PDF file produced. Values include all legal version designators (e.g., 1.2, 1.5, 2.0).
AdvancedParams ?	element	Advanced parameters that control how certain features of PDF are handled.
PDFXParams ?	element	PDF/X parameters.
ThinPDFParams ?	element	Parameters that control the optional content or form of PDF files that SHALL be created.

6.62.4 PDFXParams

Parameters for generating PDF/X files.

Note: TrimBox, BleedBox, output intent and the Trapped state may be provided by the use of the **pdfmark** operator in a PostScript source file.

Table 6.136: PDFXParams Element

NAME	DATA TYPE	DESCRIPTION
<i>PDFXBleedBoxToTrimBoxOffset</i> ?	rectangle	If the BleedBox entry is not specified in the page object of the source document, BleedBox SHALL be 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 .
<i>PDFXCheck</i> ?	NMTOKENS	List of PDF/X versions that the output SHALL be compliant with. Values include: X1a – See the PDF/X-1a standard ▶ [ISO15930-1:2001]. X3 – See the PDF/X-3 standard ▶ [ISO15930-3:2002]. X4 – See the PDF/X-4 standard ▶ [ISO15930-7:2010]. X5 – See the PDF/X-5 standard ▶ [ISO15930-8:2010].
<i>PDFXCompliantPDFOnly</i> ?	boolean	If "true", PDF document SHALL be produced only if PDF/X compliance tests are passed.
<i>PDFXNoTrimBoxError</i> ?	boolean	If "true" and both TrimBox and ArtBox entries are not specified in the page object of the source document, the condition SHALL be reported as an error.
<i>PDFXSetBleedBoxToMediaBox</i> ?	boolean	If "true" and the BleedBox entry is not specified in the page object of the source document, BleedBox SHALL be set to MediaBox .
<i>PDFXTrapped</i> ?	enumeration	If a source document does not specify a Trapped state, then the value provided here SHALL be used. The value "Unknown" SHALL 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: false true Unknown Note: "Unknown" is prohibited in PDF/X files.
<i>PDFXTrimBoxToMediaBoxOffset</i> ?	rectangle	If both the TrimBox and ArtBox entries are not specified in the page object of the source document, TrimBox SHALL be set to MediaBox with offsets. All numbers SHALL be greater than or equal to 0.0. The TrimBox SHALL be completely inside MediaBox .
<i>FileSpec</i> (ReferenceOutputProfile) ?	element	An ICC profile that describes the reference output conditions.

6.62.5 PSCreationDetails

PSCreationDetails 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 6.137: PSCreationDetails Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<code>BinaryOK</code> ?	boolean	If "false", binary data SHALL NOT be included in the PostScript stream.
<code>BoundingBox</code> ?	rectangle	<code>@BoundingBox</code> is used for the BoundingBox DSC comment in <code>@CenterCropBox</code> calculations and for PostScript's setpagedevice .
<code>CenterCropBox</code> ?	boolean	If "true", the CropBox from the source document SHALL be centered on the page when the CropBox is smaller than MediaBox .
<code>GeneratePageStreams</code> ?	boolean	If "true", the process SHALL emit individual streams of data for each page in the RunList .
<code>IgnoreAnnotForms</code> ?	boolean	If "true", annotations that contain a PDF XObject form SHALL be ignored. (PDF source only).
<code>IgnoreBG</code> ?	boolean	If "true", the BG and BG2 parameters in the PDF ExtGState dictionary, and the operand of any calls to the PostScript setblackgeneration operator SHALL be ignored.
<code>IgnoreColorSeeps</code> ?	boolean	If "true", images for Level-1 separations SHALL be ignored.
<code>IgnoreDSC</code> ?	boolean	If "true", DSC (Document Structuring Conventions) SHALL be ignored.
<code>IgnoreExternStreamRef</code> ?	boolean	If a PDF image resource uses an external stream and <code>@IgnoreExternStreamRef="true"</code> , code that points to the external file SHALL be ignored. (PDF source only).
<code>IgnoreHalftones</code> ?	boolean	If "true", any halftone screening in the source file SHALL be ignored.
<code>IgnoreOverprint</code> ?	boolean	If "true", OP parameters in a source PDF ExtGState dictionary and setoverprint in a source PostScript file, etc. SHALL be ignored.
<code>IgnorePageRotation</code> ?	boolean	If "true", a "concatenation" provided at the beginning of each page that orients the page so that it is properly rotated SHALL be ignored. Used when emitting EPS.
<code>IgnoreRawData</code> ?	boolean	If "true", no unnecessary filters SHALL be added when emitting image data.
<code>IgnoreSeparableImagesOnly</code> ?	boolean	If "true", and if emitting EPS, only CMYK and gray images SHALL be ignored.
<code>IgnoreShowPage</code> ?	boolean	If "true", save-and-restore showpage in PostScript files SHALL be ignored.
<code>IgnoreTransfers</code> ?	boolean	If "true", TR and TR2 parameters are in a source PDF ExtGState dictionary and settransfer and setcolortransfer in a source PostScript file, etc. SHALL be ignored.
<code>IgnoreTTFontsFirst</code> ?	boolean	If "true", TrueType fonts SHALL be ignored before any other fonts.
<code>IgnoreUCR</code> ?	boolean	If "true", UCR and UCR2 parameters in a source PDF ExtGState dictionary and setundercolorremoval in a source PostScript file, etc. SHALL be ignored.
<code>IncludeBaseFonts</code> ?	enumeration	Determines when to embed the base fonts. The base fonts are "Symbol" and the plain, bold, italic and bold-italic faces of "Courier", "Times" and "Helvetica". Allowed value is from: ▶ IncludeResources.
<code>IncludeCIDFonts</code> ?	enumeration	Determines when to embed CID fonts. Allowed value is from: ▶ IncludeResources.
<code>IncludeEmbeddedFonts</code> ?	enumeration	Determines when to embed fonts in the document that are embedded in the source file. Allowed value is from: ▶ IncludeResources.

Table 6.137: PSCreationDetails Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>IncludeOtherResources</i> ?	enumeration	Determines when to include all other types of resources in the file. Allowed value is from: ▶ IncludeResources.
<i>IncludeProcSets</i> ?	enumeration	Determines when to include ProcSets in the file. Allowed value is from: ▶ IncludeResources.
<i>IncludeTrueTypeFonts</i> ?	enumeration	Determines when to embed TrueType fonts. Allowed value is from: ▶ IncludeResources.
<i>IncludeType1Fonts</i> ?	enumeration	Determines when to embed Type-1 fonts. Allowed value is from: ▶ IncludeResources.
<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: EPS PostScript
<i>PSLevel</i> ?	integer	Number that indicates the PostScript level. Values include "1", "2" or "3".
<i>Scale</i> ?	float	Number that indicates the wide-scale factor of documents. Full size = "100".
<i>SetPageSize</i> ?	boolean	If "true", sets the page size on each page automatically. For PDF source, use MediaBox for outputting PostScript files and CropBox for EPS. This applies for PostScript Levels 2 and 3 only.
<i>SetupProcsets</i> ?	boolean	If "true", indicates that if ProcSets are included, the init/term code SHALL also be included.
<i>ShrinkToFit</i> ?	boolean	If "true", the page SHALL be scaled to fit the printer page size. This field SHALL override scale.
<i>SuppressCenter</i> ?	boolean	If "true", page contents whose crop box is smaller than the page size SHALL NOT be automatically centered.
<i>SuppressRotate</i>	boolean	If "true", pages with dimensions that are better suited to landscape orientation SHALL NOT be automatically rotated. 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 SHALL NOT be rotated if <i>@SuppressRotate="true"</i> . On the other hand, if <i>@SuppressRotate="false"</i> , 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 "true", and if including TrueType fonts, the fonts SHALL be converted to Type-42 rather than Type-1 fonts.
<i>UseFontAliasNames</i> ?	boolean	If "true", font alias names SHALL be used when printing with system fonts.

6.62.6 ThinPDFParams

Table 6.138: ThinPDFParams Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>FilePerPage</i> ?	boolean	If "true", the process SHALL generate 1 PDF file per page.

Table 6.138: ThinPDFParams Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>SidelineEPS</i> ?	boolean	If "true", embedded EPS files in PostScript source documents SHALL not be converted but SHALL be stored in external files in the same location as the PDF itself. (PostScript source only)
<i>SidelineFonts</i> ?	boolean	If "true", font data MAY be stored in external files during PDF generation.
<i>SidelineImages</i> ?	boolean	If "true", image data MAY be stored in an external stream during the PDF generation phase. Note: This prevents large amounts of image data from having to be passed through all phases of the code generation process.

6.63 PerforatingParams

PerforatingParams define the parameters for perforating a sheet.

Resource Properties

Intent Pairing: *FoldingIntent*

Input of Processes: *Perforating*

Table 6.139: PerforatingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>Perforate</i> +	element	Defines one or more <i>Perforate</i> lines.

6.64 PreflightParams

The *PreflightParams* resource specifies the tests for the *Preflight* process to run.

Resource Properties

Intent Pairing: *ContentCheckIntent*

Input of Processes: *LayoutElementProduction, Preflight*

Table 6.140: PreflightParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>FileSpec</i> ?	element	File that describes the preflight actions in a computer-readable, non XJDF format.
<i>PreflightTest</i> *	element	Descriptions of individual tests.

6.64.1 PreflightTest

PreflightTest describes an individual preflight test.

Table 6.141: PreflightTest Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Action</i> ?	enumeration	Action that SHOULD be taken whenever this test fails. Allowed value is from: ▶ Action.
<i>DescriptiveName</i> ?	string	Human readable description of this preflight test.
<i>Severity</i> ?	enumeration	Severity of a failure of the test. Allowed value is from: ▶ Severity. Note: If not specified, an implementation MAY generate <i>PreflightCheck/@Severity</i> based on the details of test.

Table 6.141: PreflightTest Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>TestClass</i> ?	NMTOKEN	General area of the preflight test. Values include: <i>Colorspace</i> – Tests that check color space violations. <i>FileFormat</i> – Tests that detect incorrect file format compliancy. <i>Font</i> – Tests that check font integrity. <i>PageFormat</i> – Tests that detect invalid page sizes or page boxes. <i>Resolution</i> – Tests that check low resolution graphics.
<i>TestID</i> ?	NMTOKEN	System dependent preflight test identifier.
<i>GeneralID</i> *	element	Detailed individual parameters of the <i>PreflightTest</i> . The values of <i>GeneralID</i> / <i>@IDUsage</i> and <i>GeneralID</i> / <i>@IDValue</i> are system dependent.

6.65 PreflightReport

The *PreflightReport* resource describes the results of the preflight tests specified in *PreflightParams*.

Resource Properties

Intent Pairing: *ContentCheckIntent*

Output of Processes: *Preflight*

Table 6.142: PreflightReport Resource

NAME	DATA TYPE	DESCRIPTION
<i>ErrorCount</i> ?	integer	The count of errors that were encountered while preflighting.
<i>WarningCount</i> ?	integer	The count of warnings that were encountered while preflighting.
<i>FileSpec</i> ?	element	References a human readable preflight report.
<i>PreflightCheck</i> *	element	List of individual preflight results.

6.65.1 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 6.143: PreflightCheck Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Action</i> ?	enumeration	Action that has been taken. Allowed value is from: ▶ Action.
<i>Count</i> ?	integer	The total number of occurrences of this <i>PreflightCheck</i> .
<i>Pages</i> ?	IntegerList	A 0-based index of pages in the document where one or more errors of the type specified by this <i>PreflightCheck</i> occurred.
<i>Severity</i> ?	enumeration	Severity of the <i>PreflightCheck</i> . If the <i>Preflight</i> process is described in detail with <i>PreflightTest</i> elements that include <i>PreflightTest</i> / <i>@Severity</i> , then this SHALL be a copy of the appropriate <i>PreflightTest</i> / <i>@Severity</i> . Allowed value is from: ▶ Severity.
<i>TestClass</i> ?	NMTOKEN	General area of the preflight check. If the <i>Preflight</i> process is described in detail with <i>PreflightTest</i> elements, then this SHALL be a copy of the appropriate <i>PreflightTest</i> / <i>@TestClass</i> . Allowed values are from: <i>PreflightTest</i> / <i>@TestClass</i> .
<i>TestID</i> ?	NMTOKEN	System dependent error identifier. If the <i>Preflight</i> process is described in detail with <i>PreflightTest</i> elements, then this SHALL be a copy of the appropriate <i>PreflightTest</i> / <i>@TestID</i> .
<i>Comment</i> ?	element	Human readable description of this preflight check.

Table 6.143: PreflightCheck Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>GeneralID</i> *	element	Detailed individual parameters of the <i>PreflightTest</i> . The values of <i>GeneralID</i> / <i>@IDUsage</i> and <i>GeneralID</i> / <i>@IDValue</i> are system dependent.

6.66 Preview

The preview of the content of a surface. It can be used for the calculation of ink coverage (*Part*/*@PreviewType*="Separation") or as a visual representation of what is currently processed in a device (*Part*/*@PreviewType*="Viewable" or *Part*/*@PreviewType*="ThumbNail"). When the preview is of *Part*/*@PreviewType*="Separation" or *Part*/*@PreviewType*="SeparationRaw", a gray value of "0" represents full ink, while a value of "255" represents no ink. For more information, refer to the DeviceGray color model description in the *PostScript Language Reference* ▶ [PostScript].

Resource Properties

Input of Processes: **Any Process, InkZoneCalculation**

Output of Processes: **PreviewGeneration**

Table 6.144: Preview Resource

NAME	DATA TYPE	DESCRIPTION
<i>Compensation</i> ?	enumeration	Compensation of the image to reflect the application of transfer curves to the image. Allowed value is from: ▶ Compensation.
<i>CTM</i> ?	matrix	Orientation of the <i>Preview</i> with respect to the <i>Layout</i> 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> ?	enumeration	The file type of the preview. <i>@PreviewFileType</i> SHALL NOT be specified unless the <i>Preview</i> describes an ink zone preset in CIP3 or PNG format. The file type of all other previews SHOULD be identified in <i>FileSpec</i> / <i>@MimeType</i> . Allowed values are: CIP3Multiple – The format as defined in ▶ [CIP3 - PPF]. One or more previews per CIP3 file are supported. CIP3Single – The format as defined in ▶ [CIP3 - PPF]. Only one preview per CIP3 file is supported. PNG – The Portable Network Graphics format. See ▶ [ISO/IEC 15948:2004].
<i>FileSpec</i> ?	element	<i>FileSpec</i> SHALL identify the preview (e.g., the PNG image or ▶ [CIP3 - PPF] file that represents this <i>Preview</i>).

6.67 PreviewGenerationParams

Parameters specifying the size and the type of the preview.

Resource Properties

Input of Processes: **PreviewGeneration**

Table 6.145: PreviewGenerationParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>AspectRatio</i> ?	enumeration	Policy that defines how to define the preview size if the aspect ratio of the source and preview are different. <i>@AspectRatio</i> SHALL NOT be specified unless <i>@Size</i> is also specified. Allowed values are: <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>Resolution</i> ?	XYPair	Resolution of the preview, in dpi.
<i>Size</i> ?	XYPair	Size of the preview, in pixels. If <i>@Size</i> is present, <i>@Resolution</i> SHALL be 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.

6.68 QualityControlParams

QualityControlParams defines the set of parameters for the quality control process. The specific measurement conditions SHOULD be defined in specialized subelements such as *BindingQualityParams* or as subelements of the parent *Resource* that are in a foreign namespace. Parameters for *QualityControl* MAY also be referenced by providing a *FileSpec* that references a proprietary setup definition. Examples for quality control setup in a foreign namespace include ▶ [ISO17972-1:2015] for color measurement data.

Resource Properties

Input of Processes: **QualityControl**

Table 6.146: 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.
<i>FileSpec</i> ?	element	Location of an external file that contains details of the quality control setup.

6.68.1 BindingQualityParams

The set of parameters in *BindingQualityParams* identifies how the quality of the binding is verified.

6.68.1.1 Pull test

In the pull test (sheet pulling test), a single sheet is subjected to slowly increasing tensile loading until it comes away from the glue film or the material breaks down. The load increases constantly during the automatic test procedure. It is applied evenly along the whole length of the glued seam.

Note: That is why the pull test is also described as a static test method.

6.68.1.2 Flex test

The page flex test (page turning test) is used more and more rarely in quality checking, not least because it is time consuming. In the page flex test a sheet is moved back and forth under varying tensile loads, usually at 1 N/cm, until it pulls out of the glue film, with the number of to and fro movements being measured automatically.

Note: As this test procedure involves a rapid turning movement, the flex test is called a dynamic test procedure.

Table 6.147: BindingQualityParams Element

NAME	DATA TYPE	DESCRIPTION
<i>FlexValue</i> ?	float	Flex quality parameter measured in [N/cm].
<i>PullOutValue</i> ?	float	Pull out quality parameter measured in [N/cm].

6.69 QualityControlResult

QualityControlResult defines the set of results from a **QualityControl** process. The specific measurements should be returned in specialized subelements such as **BindingQualityMeasurement** or as subelements of the parent **Resource** that are in a foreign namespace. **QualityControl** results may also be referenced by providing a **FileSpec** that references a proprietary measurement results definition. Examples for quality control measurements in a foreign namespace include ▶ [ISO17972-1:2015] for color measurement data.

Resource Properties

Output of Processes: **QualityControl**

Table 6.148: QualityControlResult Resource

NAME	DATA TYPE	DESCRIPTION
<i>End</i> ?	dateTime	Date and time of the end of the measurement. If not specified, the value of <i>@Start</i> is applied.
<i>Failed</i> ?	integer	Total number of failed measurements.
<i>Passed</i> ?	integer	Total number of passed measurements.
<i>Start</i> ?	dateTime	Date and time of the start of the measurement.
<i>BindingQualityMeasurement</i> *	element	Individual measurement results of binding quality.
<i>FileSpec</i> ?	element	Location of an external file that contains details of the quality control measurement.

6.69.1 BindingQualityMeasurement

Table 6.149: BindingQualityMeasurement Element

NAME	DATA TYPE	DESCRIPTION
<i>FlexValue</i> ?	float	Flex quality parameter given in [N/cm].
<i>PullOutValue</i> ?	float	Pull out quality parameter given in [N/cm].

6.70 RasterReadingParams

This set of parameters specifies the details for **RasterReading**.

Resource Properties

Input of Processes: **RasterReading**

Table 6.150: RasterReadingParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Center</i> ?	boolean	Indicates whether or not the finished page image SHALL be centered within the imagable area of the media. <i>@Center</i> SHALL NOT be specified if <i>FitPolicy/@SizePolicy="ClipToMaxPage"</i> .

Table 6.150: RasterReadingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>FilmRef</i> ?	IDREF	Reference to film <i>Media</i> . This resource provides a description of the physical media that will be marked. The physical characteristics of the media MAY affect decisions made during <i>RasterReading</i> .
<i>MirrorAround</i> ?	enumeration	This attribute specifies the axis around which a raster reader SHALL mirror an image. Allowed value is from: ▶ Axis.
<i>PaperRef</i> ?	IDREF	Reference to final paper <i>Media</i> . This resource provides a description of the physical media that will be marked. The physical characteristics of the media MAY affect decisions made during <i>RasterReading</i> .
<i>PlateRef</i> ?	IDREF	Reference to plate <i>Media</i> . This resource provides a description of the physical media that will be marked. The physical characteristics of the media MAY affect decisions made during <i>RasterReading</i> .
<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 value is from: ▶ Polarity.
<i>ProofPaperRef</i> ?	IDREF	Reference to paper <i>Media</i> used for proofing. This resource provides a description of the physical media that will be marked. The physical characteristics of the media MAY affect decisions made during <i>RasterReading</i> .
<i>Scaling</i> ?	XYPair	A pair of positive real values that indicates the scaling factor for the content. Values between 0 and 1 specify that the contents SHALL be reduced, while values greater than 1 specify that the contents SHALL be expanded. Any scaling defined in <i>FitPolicy</i> 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 SHALL 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 "0 0".
<i>FitPolicy</i> ?	element	Allows printing even if the size of the imaggable area of the media does not match the requirements of the data.

6.71 RenderingParams

This set of parameters identifies how the *Rendering* process SHALL operate. Specifically, these parameters define the expected output of the *RunList* that the *Rendering* process creates.

Resource Properties

Input of Processes: *Rendering*

Table 6.151: 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.
<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.

Table 6.151: RenderingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ColorantDepth</i> ?	integer	Number of bits per colorant. Determines whether the output is bitmaps or bytemaps.
<i>Interleaved</i> ?	boolean	If "true", the resulting colorant values SHALL be interleaved. @BandOrdering SHALL NOT be specified if @Interleaved="true".
<i>MimeType</i> ?	string	@MimeType identifies the MIME type associated with this output file format. For example "application/pdf".
<i>AutomatedOverprintParams</i> ?	element	Controls for overprint substitutions. Defaults to no automated overprint generation.
<i>ObjectResolution</i> *	element	Elements that define the resolutions at which to render the contents. More than one element MAY be used to specify different resolutions for different @SourceObjects types. If no ObjectResolution is specified, the value is implied from the input data.
<i>TIFFFormatParams</i> ?	element	Parameters specific for creating TIFF files.

6.71.1 TIFFEmbeddedFile

Table 6.152: 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 that SHALL be embedded.

6.71.2 TIFFFormatParams

Table 6.153: TIFFFormatParams Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ByteOrder</i> ?	enumeration	Byte order of the TIFF file. Allowed values are: ll – Low byte first. MM – 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—PlanarConfiguration: Allowed values are: 1 – “Chunky” format, which is pixel interleaved. 2 – “Planar” format, which is strip interleaved.
<i>RowsPerStrip</i> ?	integer	The number of image scan lines per strip, encoded in the TIFF file as RowsPerStrip. This attribute is ignored if @Segmentation!="Stripped". The default, if not known, is set by the processing system with the exception that when converting from ByteMap to TIFF, ByteMap/Band/@Height is the default.

Table 6.153: TIFFFormatParams Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Segmentation</i> ?	enumeration	How the image data are segmented. Allowed values are: SingleStrip – All data are included in one segment. This is encoded in the TIFF file by setting RowsPerStrip to a number equal to or larger than the number of pixel rows in the image. Stripped – Data are segmented into strips. Tiled – 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.
<i>TileSize</i> ?	XYPair	Two integers. The X value provides width of tiles, and the Y value provides height of tiles. <i>@TileSize</i> SHALL NOT be specified unless <i>@Segmentation</i> ="Tiled".
<i>WhitelsZero</i> ?	boolean	When writing monochrome or gray scale files, this flag indicates whether the data SHALL be written with either white values encoded as zero, "true" or black values encoded as zero, "false".
<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.

6.71.3 TIFFtag

Table 6.154: 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> ?	FloatList	If the type of the tag is RATIONAL, SRATIONAL or FLOAT, 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, Copy-right, DateTime, ImageDescription, etc.

6.72 RunList

A **RunList** defines one or more printable logical documents or document sets that MAY be defined in one or more external physical PDL or image files. It retains the properties of the original documents, e.g. the pages of a set of documents with ordered pages that are described by a **RunList**, retain that order.

RunList allows structuring of multiple pages into documents. Multiple documents that have a joint context may be grouped into sets. For examples of how these can be mapped see ▶ Section 6.72.3 Pages, Documents and Sets for common PDL types.

6.72.1 Referencing pages of a RunList from a Layout

The **Layout** resource in the **Imposition** process references individual pages in a **RunList** by index in **Layout/@Ord**. The index SHALL be calculated in the context of the current document and set of a **RunList**.

6.72.2 Filtering parts of a RunList

The partition keys: **Part/@DocIndex**, **Part/@RunIndex**, **Part/@SetIndex** and **Part/@SheetIndex** are provided to select subsets of a **RunList** without modifying the indexes of selected pages for purposes of calculating **@Ord** values in **Imposition**. These keys SHALL be provided in the **ResourceSet/@Usage="Output"** of the process that requires filtering, e.g. for selective reprint of a set of pages, sheets, documents or document sets.

The following example selects the first page (from two pages) of 'file1.pdf' and the second page through to the fifth page (from eight pages) of 'file2.pdf'. The first **Part** of the output **RunList**, with **@RunIndex="0 0"**, selects the range from first to first page of the entire input **RunList**. The second **Part** of the output **RunList**, with **@RunIndex="3 6"**, selects the range from fourth to seventh page of the entire input **RunList**. Since the first input **RunList** partition contains two pages, the fourth through seventh page are the second through fifth page of the second input **RunList** partition.

Example 6.18: Filtering Parts of a RunList

```
<ResourceSet Name="RunList" Usage="Input">
  <Resource>
    <Part Run="r1"/>
    <RunList NPage="2">
      <FileSpec URL="file:///indir/file1.pdf"/>
    </RunList>
  </Resource>
  <Resource>
    <Part Run="r2"/>
    <RunList NPage="8">
      <FileSpec URL="file:///outdir/output.pdf"/>
    </RunList>
  </Resource>
</ResourceSet>
<ResourceSet Name="RunList" Usage="Output">
  <Resource>
    <Part RunIndex="0 0"/>
    <Part RunIndex="3 6"/>
    <RunList/>
  </Resource>
</ResourceSet>
```

Resource Properties

Resource referenced by: **Layout/SheetActivation**

Input of Processes: **ColorCorrection, ColorSpaceConversion, DigitalPrinting, ImageSetting, Imposition, Interpreting, LayoutElementProduction, LayoutShifting, PDLCreation, Preflight, PreviewGeneration, RasterReading, Rendering, Screening, Separation, ShapeDefProduction, Stripping, Trapping**

Output of Processes: **ColorCorrection, ColorSpaceConversion, Imposition, Interpreting, LayoutElementProduction, LayoutShifting, PDLCreation, RasterReading, Rendering, Screening, Separation, Stripping, Trapping**

Table 6.155: RunList Resource (Sheet 1 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>Automation</i> ?	enumeration	Identifies dynamic and static RunList elements. Generating Resource/Part in automated imposition is defined in detail in ▶ Section 5.4.8.1 Execution Model for Automated Imposition. This structure SHALL be retained in the RunList description. If @Automation="Dynamic" and ResourceSet/Dependent/@PipeID is also present, details MAY be specified in XJMF pipe messages. See ▶ Section 9.3.5.1 Dynamic Pipes. Allowed value is from: ▶ Automation.

Table 6.155: RunList Resource (Sheet 2 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>ClipPath</i> ?	PDFPath	Path that describes the outline of the RunList that SHALL be clipped. The default case is that there is no clip path. @ClipPath , @SourceClipBox , PlacedObject / @SourceClipPath and PlacedObject / @ClipBox , 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.
<i>Docs</i> ?	IntegerRange	Zero-based range of document indices in a multi-document file in the context of @Sets that SHALL be selected in the order of the range. If not present, all documents SHALL be selected.
<i>EndOfDocument</i> ?	boolean	If "true", the last page in the RunList is the last page of an instance document. If the RunList references a PDL that supports internal instance documents, @EndOfDocument SHALL be the value that is defined in the PDL. The implied default value of @EndOfDocument ="false", except for the last RunList Resource with the same explicit or calculated value of Part / @SetIndex , which has an implied default value of @EndOfDocument ="true".
<i>EndOfSet</i> ?	boolean	If "true", the last page in the RunList is the last page of a set of instance documents. If the RunList references a PDL that supports internal sets, @EndOfSet SHALL be the value that is defined in the PDL. The implied default value of @EndOfSet ="false", except for the last RunList Resource , which has an implied default value of @EndOfSet ="true".
<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>LogicalPage</i> ?	integer	The logical page number of the first page in a RunList . It defaults to "1" plus the last page of the previous sibling RunList partition. If the RunList resource is the first partition, @LogicalPage defaults to "0". @LogicalPage SHALL NOT be specified lower than the highest calculated value of @LogicalPage of a previous partition.
<i>NPage</i> ?	integer	Total number of pages (placed object slots) that are defined by the RunList . If @NPage is not specified, it defaults to all pages in the referenced PDL. If the RunList describes multiple instance documents or document sets, @NPage refers to the total number of pages in all instance documents and sets. A RunList with @NPage specified always refers to @NPage pages, regardless of the number of pages of the referenced PDL.
<i>OrdType</i> ?	enumeration	@OrdType SHALL specify the usage of this RunList element in the context of an Imposition process. Allowed values are: Content - This RunList specifies a set of pages that have content associated. @NPage SHOULD be specified. Insert - This RunList specifies a set of pages that will be inserted from an external feeder. @NPage SHALL be specified. Reservation - This RunList specifies a set of pages that have no content associated. @NPage SHALL be specified.
<i>Pages</i> ?	IntegerRange	Zero-based range of indices of the pages in the context of @Docs and @Sets that SHALL be selected in the order of the range. If neither @Pages nor @NPage is specified, all pages in the PDL referred to by the RunList SHALL be selected in document order.
<i>Sets</i> ?	IntegerRange	Zero based range of document-set indices in a multi document-set file specified by the RunList that SHALL be selected in the order of the range. If not present, all document sets SHALL be selected.
<i>SourceBleedBox</i> ?	rectangle	A rectangle that describes the bleed area of the element that SHALL be included. This rectangle is expressed in the source coordinate system of the object.

Table 6.155: RunList Resource (Sheet 3 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>SourceClipBox</i> ?	rectangle	A rectangle that defines the region of the element that SHALL be included. This rectangle is expressed in the source coordinate system of the object.
<i>SourceMediaBox</i> ?	rectangle	A rectangle that defines the intended media size of the element. 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 element to be included. This rectangle is expressed in the source coordinate system of the object.
<i>ByteMap</i> ?	element	Describes the page or stream of pages. At most one of <i>ByteMap</i> or <i>FileSpec</i> SHALL be specified. If none of <i>ByteMap</i> or <i>FileSpec</i> are specified, the <i>RunList</i> specifies empty content.
<i>FileSpec</i> ?	element	URL plus metadata about the physical characteristics of a file representing the <i>RunList</i> . If not present, then only metadata is known but not the content file.
<i>MetadataMap</i> *	element	Describes the mapping of metadata in a <i>RunList</i> to partition keys. <i>MetadataMap</i> SHOULD NOT be specified unless <i>@Automation</i> ="Dynamic".

6.72.3 Pages, Documents and Sets for common PDL types

The following table defines the mapping of *RunList* structures to commonly used PDLs.

Table 6.156: Pages, Documents and Sets for common PDL types

PDL	PAGES	DOCUMENTS	SETS	REMARKS
Post-Script	-	-	PostScript is a single document PDL	
PDF	Page in pages tree	-	-	Regular PDF including PDF/X is a single document PDL
PDF/VT	Page in pages tree	Any DPart Descendant below the Set	Any DPart record as defined by RecordLevel	A Record as defined by Record-Level SHALL be mapped to a Set
PPML	PAGE elements	DOCUMENT elements	DOCUMENT_SET/JOB elements	

6.72.4 Band

Table 6.157: Band Element

NAME	DATA TYPE	DESCRIPTION
<i>Height</i> ?	integer	Height in pixels of the band.
<i>Width</i> ?	integer	Width in pixels of the band.

6.72.5 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 6.158: ByteMap Element

NAME	DATA TYPE	DESCRIPTION
<i>BandOrdering</i> ?	enumeration	Identifies the precedence given when ordering the produced bands. @ <i>BandOrdering</i> SHALL be specified 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, e.g. in a previous Screening process.
<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 identifiers containing information about which colorants are represented. Additional details of the colorants SHOULD be provided in <i>ResourceSet</i> [@Name="Color"].
<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 of the <i>ByteMap</i> in dpi.
<i>Band</i> ?	element	Description of the structure of the bands or tiles containing the raster data.

Example 6.19: RunList: Unstructured Single-File RunList

The order in which the *RunList* elements appear in the XML document is significant. **Note:** The @*Run* partition key has a string value, which MAY be non-numeric. Below is an example of a simple unstructured single-file *RunList*. This example specifies all pages contained in "/in/colortest.pdf".

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="RunList" Types="Imposition">
  <ResourceSet Name="RunList" Usage="Input">
    <Resource>
      <RunList Pages="0 -1">
        <FileSpec URL="File:///in/colortest.pdf"/>
      </RunList>
    </Resource>
  </ResourceSet>
</XJDF>
```

6.73 ScreeningParams

ScreeningParams specifies the parameters of the **Screening** process.

Resource Properties

Input of Processes: **Screening**

Table 6.159: ScreeningParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>IgnoreSourceFile</i> ?	boolean	If @ <i>IgnoreSourceFile</i> ="true", the screen settings specified in a source file SHALL NOT be applied, e.g. setscreen , setcolorscreen and sethalftone in a PDF file.

Table 6.159: ScreeningParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
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.

6.74 SeparationControlParams

[SeparationControlParams](#) provides the controls needed to separate composite color files.

Resource Properties

Input of Processes: [Separation](#)

Table 6.160: SeparationControlParams Resource

NAME	DATA TYPE	DESCRIPTION
AutomatedOverPrintParams ?	element	Controls for overprint substitutions.

6.75 ShapeCuttingParams

[ShapeCuttingParams](#) defines the details of the [ShapeCutting](#) process.

Resource Properties

Intent Pairing: [ShapeCuttingIntent](#)

Input of Processes: [ShapeCutting](#)

Table 6.161: ShapeCuttingParams Resource

NAME	DATA TYPE	DESCRIPTION
DeliveryMode ?	enumeration	Allowed values are: FullSheet – The output of the die-cutter SHALL be complete sheets. The blanks are kept in place with nicks. Front waste (gripper margin) SHALL NOT be removed. RemoveGripperMargin – The output of the die-cutter SHALL be complete sheets. The blanks are kept in place with nicks. Front waste (gripper margin) SHALL be removed. SeparateBlanks – The output of the die-cutter SHALL be blanks that have been removed from the sheets.
DieLayoutRef ?	IDREF	Reference to a DieLayout containing the reference of an external file describing the cutting and other paths.
ModuleID ?	NMTOKEN	Identifier of the shape-cutting module in a multi-function device, such as a printing press. See Module for details.
SheetLay ?	enumeration	Lay of input media. Reference edge of where the sheets are placed in the feeder. Allowed value is from: ▶ SheetLay .
Shape *	element	Details of each individual cut shape.

6.76 ShapeDef

A structural design describing a 2D surface with paths that describe different finishing operations such as 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.

Resource Properties

Resource referenced by: [DieLayout/Station](#), [DieLayoutProductionParams/RepeatDesc](#), [LayoutElementProductionParams](#)

Output of Processes: **ShapeDefProduction**

Table 6.162: ShapeDef Resource

NAME	DATA TYPE	DESCRIPTION
<i>Area</i> ?	float	The net area of the shape in m ² , after cutting.
<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 a blank box.
<i>CutLines</i> ?	NMTOKENS	Selects the die line separation identifiers from the file referenced by <i>FileSpec</i> . Additional details of the usage of the separations MAY be specified in the respective <i>ResourceSet</i> [@Name="Color"].
<i>Dimensions</i> ?	shape	Width x, height y and depth z 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 @FlatDimensions.
<i>FlatDimensions</i> ?	shape	Width x, height y and depth z 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 <i>Component</i> /@Dimensions of the output 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 SHALL be taken into account by the DieLayoutProduction process to give the <i>ShapeDef</i> the correct orientation on the sheet. Allowed value is from: ▶ MediaDirection.
<i>GrainDirection</i> ?	enumeration	Intended direction of the grain for this design in the coordinate system defined by @CutBox. This information SHALL be taken into account by the DieLayoutProduction process to give the <i>ShapeDef</i> the correct orientation on the sheet. Allowed value is from: ▶ MediaDirection.
<i>MediaRef</i> ?	IDREF	Reference to a <i>Media</i> resource for which this structural design was intended. The <i>Media</i> description defines important design parameters, such as the type of <i>Media</i> , thickness, inside loss, outside gain, etc.
<i>MediaSide</i> ?	enumeration	Determines the printing side for which the DieLayout is made. "Front" corresponds to the outside of a box, "Back" corresponds to the inside of a box. Allowed value is from: ▶ Side. Note: Folding carton is usually cut from the outside (Front), corrugated from the inside (Back).
<i>ResourceWeight</i> ?	float	The weight of the shape after cutting (g).
<i>FileSpec</i> ?	element	The <i>FileSpec</i> of the structural design file. The format of this file may be a vendor specific format, a standard format like ▶ [DDES3], a less well specified but commonly used format like CFF2 or DXF or even a PDF or EPS file. <i>FileSpec</i> and <i>Shape</i> are mutually exclusive.
<i>Shape</i> *	element	The shape is defined by a collection of <i>Shape</i> elements. <i>Shape</i> and <i>FileSpec</i> are mutually exclusive.

6.77 ShapeDefProductionParams

Parameters for the structural design.

Resource Properties

Input of Processes: **ShapeDefProduction**

Table 6.163: ShapeDefProductionParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ObjectModel</i> *	element	A 3D model of the objects that need to be packed.

Table 6.163: ShapeDefProductionParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
ShapeTemplate ?	element	A structural template sometimes referred to as a parametric structural design. Given a set of parametric values, a structural template can be instantiated to an actual structural design.

6.77.1 ObjectModel

Table 6.164: ObjectModel Element

NAME	DATA TYPE	DESCRIPTION
Dimensions ?	shape	Width x, height y and depth z 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).

6.77.2 ShapeDimension

Table 6.165: ShapeDimension Element

NAME	DATA TYPE	DESCRIPTION
Usage	string	@Usage specifies the name of the ShapeDimension . Values Include: D - Depth L - Length W - Width
Value	float	@Value specifies the length of the ShapeDimension in points.

6.77.3 ShapeTemplate

[ShapeTemplate](#) describes a structural template that is also referred to as a parametric structural design.

Table 6.166: ShapeTemplate Element

NAME	DATA TYPE	DESCRIPTION
InnerDimensions ?	shape	Width x, height y and depth z coordinates of the 3D shape. For a box these are the inner dimensions.
Name ?	NMTOKEN	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 6.20: ShapeTemplate for template example 1 below illustrates the encoding of L, D and W using [ShapeDimension](#).

Example 6.20: ShapeTemplate for template example 1

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="n_001010"
  JobPartID="n_000006" Types="ShapeDefProduction">
  <ResourceSet Name="ShapeDefProductionParams" Usage="Input">
    <Resource DescriptiveName="Box123">
      <ShapeDefProductionParams>
        <ShapeTemplate
          InnerDimensions="566.92913386 708.66141732 283.46456693" Standard="ECMA">
          <ShapeDimension Usage="L" Value="566.93"/>
          <ShapeDimension Usage="W" Value="283.46"/>
          <ShapeDimension Usage="D" Value="708.66"/>
        </ShapeTemplate>
      </ShapeDefProductionParams>
    </Resource>
  </ResourceSet>
</XJDF>
```

Figure 6-33: ShapeTemplate example 1

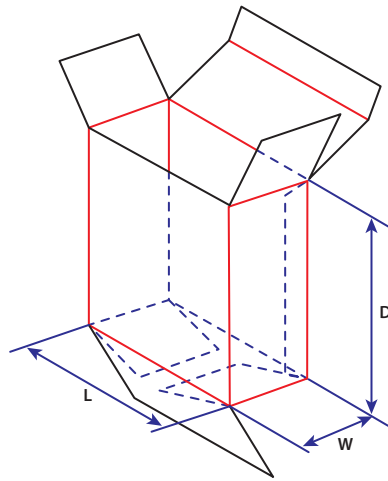


Figure 6-34: ShapeTemplate example 2

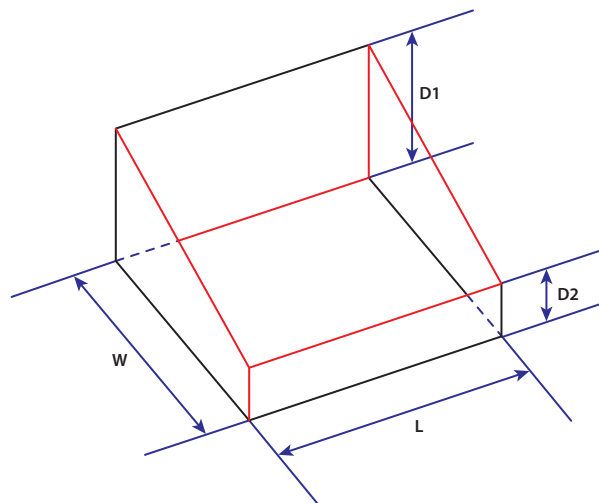
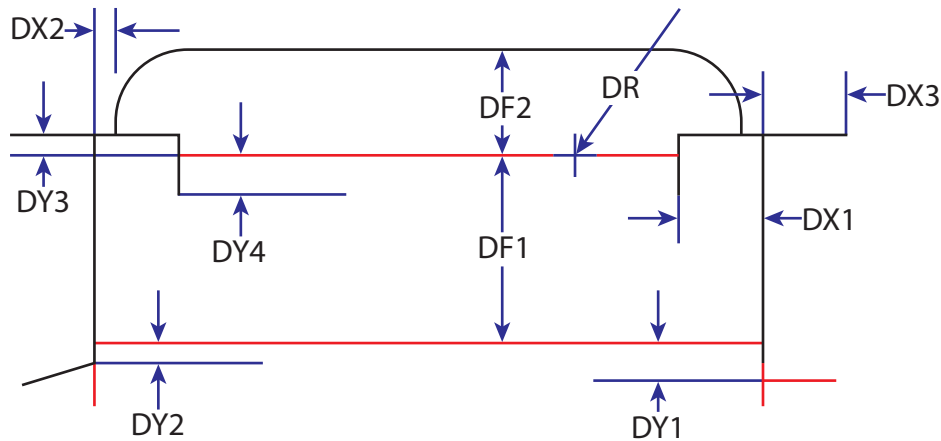


Figure 6-35: ShapeTemplate example 3



6.78 SheetOptimizingParams

SheetOptimizingParams describes the parameter set for the **SheetOptimizing** process.

Resource Properties

Input of Processes: **SheetOptimizing**

Table 6.167: 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.

6.78.1 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 6.168: GangElement Element (Sheet 1 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>BinderySignatureIDs</i> ?	NMTOKENS	If specified, <i>@BinderySignatureIDs</i> SHALL list <i>Part/@BinderySignatureID</i> of all <i>BinderySignatures</i> that are candidates for this GangElement .
<i>CollapseBleeds</i> ?	boolean	If a single page GangElement that has a bleed in a solid color is ganged in a 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. <i>@Dimension</i> SHALL NOT be specified if either <i>@BinderySignatureIDs</i> , <i>@NPage</i> or <i>@PageDimension</i> is 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 <i>@DueDate</i> and <i>@Priority</i> to decide which GangElement elements to place on a gang.
<i>ExternalID</i> ?	NMTOKEN	The ID of the product in an external system, e.g. a web to print management system.

Table 6.168: GangElement Element (Sheet 2 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>FillPriority</i> ?	integer	If non-zero, the ganging engine is requested to fill any left over space on the sheet with this GangElement element 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.
<i>GangElementID</i>	NMTOKEN	An identifier of the GangElement that is unique within the context of the workflow. <i>@GangElementID</i> SHALL be copied to the output Layout/Position/@GangElementID to indicate which GangElements have been included in the results of the SheetOptimizing 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 value is from: ▶ MediaDirection .
<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 printing on an offset press.
<i>JobID</i> ?	NMTOKEN	The original <i>@JobID</i> of the element to be ganged.
<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	Reference to a Media resource whose characteristics 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 . <i>@NPage</i> SHALL NOT be specified if <i>@BinderySignatureIDs</i> is 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 SHALL be placed on the gang in a single block. If Y is zero, then X SHALL specify the total number-up requested without specifying a specific number in the X or Y direction.
<i>OneSheet</i> ?	NMTOKEN	<i>@OneSheet</i> controls how this GangElement SHOULD be placed on ganged sheets. Values include: Any – Place on any sheet that is generated. GangElementID – Keep all blocks with this <i>@GangElementID</i> on one sheet. JobID – Keep all GangElement elements with the same <i>@JobID</i> on the same sheet.
<i>OrderQuantity</i>	integer	The number of printed (fold) sheets to produce, including finishing waste.
<i>PageDimension</i> ?	XYPair	The page size, including trims and bleeds, of the element to be ganged. <i>@PageDimension</i> SHALL NOT be specified if <i>@NPage</i> is NOT specified or if either <i>@BinderySignatureIDs</i> or <i>@Dimension</i> is specified.
<i>Priority</i> ?	integer	<i>@Priority</i> controls the relative order of including GangElement items in a gang. The value of <i>@Priority</i> SHALL be between "0" and "100". All GangElement elements with a <i>@Priority</i> ="100" SHALL be included in the gang. GangElement elements with a <i>@Priority</i> value less than 100 MAY be included in the gang, and SHOULD be included 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 value is from: ▶ PositionPolicy .

Table 6.168: GangElement Element (Sheet 3 of 3)

NAME	DATA TYPE	DESCRIPTION
<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.
<i>SeparationListBack</i> ?	NMTOKENS	<i>@SeparationListBack</i> SHALL specify the list of separation identifiers that are required for the back side of the product. MAY include varnish. Additional details of the colorants SHOULD be provided in ResourceSet [<i>@Name="Color"</i>].
<i>SeparationListFront</i> ?	NMTOKENS	<i>@SeparationListFront</i> SHALL specifies the list of separation identifiers that are required for the front side of the sheet. MAY include varnish. Additional details of the colorants SHOULD be provided in ResourceSet [<i>@Name="Color"</i>].

6.79 ShrinkingParams

ShrinkingParams provides the parameters for the **Shrinking** process in shrink wrapping.

Resource Properties

Input of Processes: **Shrinking**

Table 6.169: ShrinkingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>ShrinkingMethod</i> ?	enumeration	Specifics of the shrinking method for shrink wrapping. Allowed values are: ShrinkCool ShrinkHot
<i>Temperature</i> ?	float	Shrinking temperature.

6.80 SpinePreparationParams

SpinePreparationParams describes the preparation of the spine of book blocks for hard and softcover book production (e.g., milling and notching).

Resource Properties

Intent Pairing: **BindingIntent**

Input of Processes: **SpinePreparation**

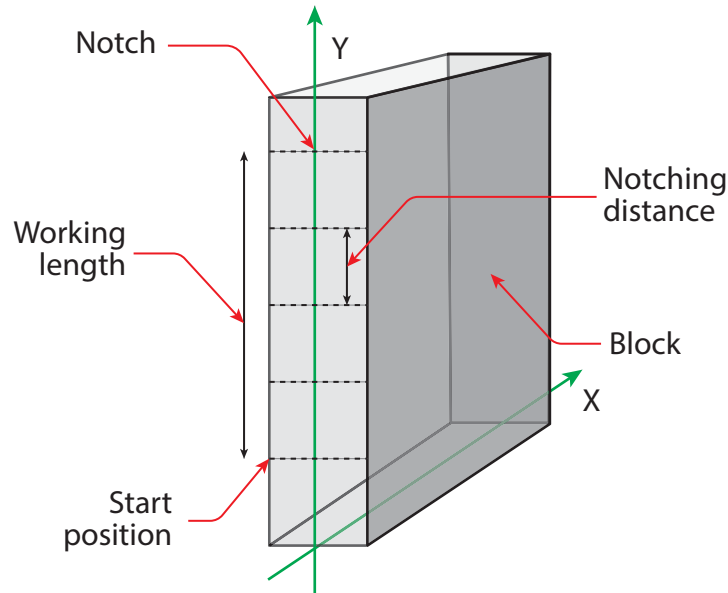
Table 6.170: SpinePreparationParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>MillingDepth</i> ?	float	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> ?	float	Notching depth relative to the leveled spine, in points. If not specified, no notching SHALL be performed.
<i>NotchingDistance</i> ?	float	Notching distance, in points.
<i>Operations</i> ?	NMTOKENS	List of operations that SHALL be applied to the spine. Duplicate entries SHALL specify a sequence of identical operations. The order of operations is significant. Values include those from: ▶ Spine Operations.
<i>SealingTemperature</i> ?	integer	<i>@SealingTemperature</i> is the temperature needed to melt the sealing thread and sheet, thereby gluing the signatures together. <i>@SealingTemperature</i> SHALL NOT be specified unless <i>@Operations</i> contains "Sealing".
<i>StartPosition</i> ?	float	Starting position of the milling tool along the Y-axis of the operation coordinate system.

Table 6.170: SpinePreparationParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>WorkingLength</i> ?	float	Working length of the milling operation. If not specified, the complete spine SHALL be prepared.

Figure 6-36: Parameters and coordinate systems for the SpinePreparation process



6.81 SpineTapingParams

SpineTapingParams define the parameters for taping a strip tape or kraft paper to the spine of a book block.

Resource Properties

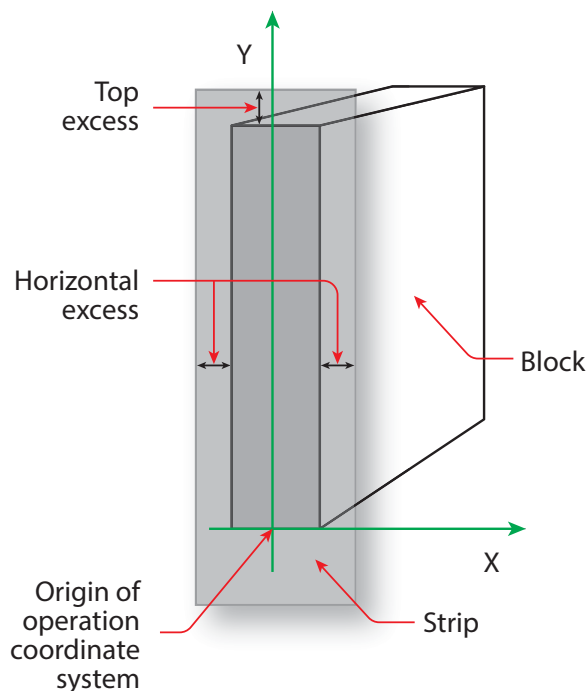
Intent Pairing: *BindingIntent*

Input of Processes: *SpineTaping*

Table 6.171: SpineTapingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>HorizontalExcess</i> ?	float	Taping spine excess on each side. The tape is assumed to be centered between left and right.
<i>HorizontalExcessBack</i> ?	float	Horizontal excess of back if tape is not centered.
<i>StripLength</i> ?	float	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>TopExcess</i> ?	float	Top spine taping excess. This value MAY be negative.
<i>Glue</i> *	element	Describes where and how to apply glue to the book block.

Figure 6-37: Parameters and coordinate system for the SpineTaping process



6.82 StackingParams

Settings for the **Stacking** process.

Resource Properties

Input of Processes: **Stacking**

Table 6.172: StackingParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>BundleType</i> ?	enumeration	@ <i>BundleType</i> specifies the <i>BundleItem</i> / <i>@BundleType</i> of the items that shall be counted as an individual item for Stacking . Allowed values are from: ▶ <i>BundleType</i> .
<i>Compensate</i> ?	boolean	180 degree rotation applied to successive layers to compensate for uneven stacking. If <i>@LayerAmount</i> = <i>@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> . If not specified, the default case is that the value of <i>@LayerAmount</i> equals the value of <i>@StandardAmount</i> .
<i>LayerCompression</i> ?	boolean	If <i>@LayerCompression</i> ="true", layer is compressed before next layer is started.
<i>LayerLift</i> ?	boolean	If <i>@LayerLift</i> ="true", layer is lifted to reduce height.
<i>MaxAmount</i> ?	integer	Maximum number of products in a stack, <i>@MaxAmount</i> SHALL be greater than or equal to <i>@StandardAmount</i> . If not specified, the default case is that the value of <i>@MaxAmount</i> equals the value of <i>@StandardAmount</i> .
<i>MaxHeight</i> ?	integer	Maximum height of the stack in points.
<i>MaxWeight</i> ?	float	Maximum weight of a stack in grams.
<i>MinAmount</i> ?	integer	Minimum number of products in a stack or layer, i.e. where (<i>@MaxAmount</i> – <i>@StandardAmount</i>) <= <i>@MinAmount</i> < <i>@StandardAmount</i> and <i>@MinAmount</i> < <i>@LayerAmount</i> . Where not specified, the default case SHALL use a value equal to <i>@MaxAmount</i> – <i>@StandardAmount</i> .

Table 6.172: StackingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>OutputBin</i> ?	NMTOKENS	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 <i>@StackAmount</i> . Values include those from: ▶ Input Tray and Output Bin Names.
<i>PreStackAmount</i> ?	integer	Amount that is initially gathered.
<i>PreStackMethod</i> ?	enumeration	Allowed values are: All – All layers are pre-stacked. First – Only first layer is pre-stacked. None – No pre-stacking.
<i>StackAmount</i> ?	integer	Specifies the maximum sheet count before switching to the next stacker in the list of <i>@OutputBin</i> values.
<i>StackCompression</i> ?	boolean	If <i>@StackCompression</i> ="true", the 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 <i>@UnderLays</i> . If less layers than values are specified, all underlay sheets that are not adjacent to a layer SHALL be ignored.
<i>Disjointing</i> ?	element	Details of the offset or shift applied to successive layers or documents to separate the thicker portions of components, for example, offsetting the spines of hardcover books.

6.82.1 Disjointing

Disjointing describes how individual items as specified by *@BundleType* are separated from one another on a stack or in an output bin or stacker. *Disjointing* SHALL apply only to the current job and SHALL NOT apply to jobs that follow.

Note: A less granular disjointing implies more granular disjointing, e.g. a set break implies a document break and a sheet break.

Table 6.173: Disjointing Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Amount</i> ?	integer	The number of components that SHALL be shifted in <i>@Direction</i> simultaneously. <i>@Units</i> SHALL specify the type of the component counted by this attribute. Any less granular component break than the value of <i>@Units</i> SHALL cause a shift regardless of the value of <i>@Amount</i> , e.g. a set break would cause a shift even if the current number of documents were less than <i>@Amount</i> in the case where <i>@Units</i> ="Docs" or <i>@Units</i> ="DocCopies".
<i>Direction</i> ?	enumeration	Offset-shift action for the first component. A component can be offset to one of two positions—left or right. Allowed values are: Alternate – 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. Left – The first component of a new job is on the left, and subsequent components are each offset to alternating positions. None – Do not offset consecutive components. The position of all components is the same as the position of the previous component. Right – The first component of a new job is on the right, and subsequent components are each offset to alternating positions.
<i>Offset</i> ?	XYPair	Offset dimension in X and Y dimensions that separates the components.

Table 6.173: Disjoining Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Units</i> ?	NMTOKEN	This attribute specifies the type of component counted by the <i>@Amount</i> attribute. Values include: <i>DocCopies</i> - Every individual document is counted. <i>Docs</i> - All copies of identical documents are counted as one. <i>Jobs</i> - The entire job SHALL be counted. For this case <i>@Amount</i> SHALL have a value of 1, i.e. when <i>@Units</i> ="Jobs", <i>@Amount</i> ="1". <i>SetCopies</i> - Every individual set is counted. <i>Sets</i> - All copies of identical sets are counted as one. <i>Sheets</i> - Every individual sheet is counted.
<i>InsertSheet</i> ?	element	Some kind of physical marker (e.g., a paper strip or a yellow paper sheet) that separates the components.

6.82.2 InsertSheet

InsertSheet resources define device generated sheets that SHALL be produced along with the job. *InsertSheet* elements include separator sheets, error sheets, accounting sheets and job sheets. The information provided on the sheet depends on the type of sheet.

Table 6.174: InsertSheet Element

NAME	DATA TYPE	DESCRIPTION
<i>IsWaste</i> ?	boolean	Specifies whether the <i>InsertSheet</i> is waste. If "true", the <i>InsertSheet</i> SHALL be discarded when finishing the document.
<i>SheetFormat</i> ?	NMTOKEN	Identifies that device dependent information SHALL be included on the <i>InsertSheet</i> . <i>@SheetFormat</i> MAY specify site dependent or customer dependent values. Values include: <i>Blank</i> - Empty sheet. <i>Brief</i> - Site specific sheet with minimal information. <i>Full</i> - Site specific sheet with maximum information. <i>Standard</i> - Site specific sheet with default information.
<i>SheetType</i>	enumeration	Identifies the type of sheet. Allowed values are: <i>AccountingSheet</i> - A sheet that reports accounting information for the job. <i>ErrorSheet</i> - A sheet that reports errors for the job. <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.
<i>SheetUsage</i>	enumeration	Indicates where this <i>InsertSheet</i> SHALL be produced and inserted into the set of output pages. Allowed values are from: ▶ Table 6.175 SheetUsage Attribute Values.
<i>StripMark</i> *	element	<i>StripMark</i> provides formatting and content for the <i>InsertSheet</i> .

Table 6.175: SheetUsage Attribute Values (Sheet 1 of 2)

VALUE	DESCRIPTION
Header	Valid for <i>@SheetType</i> ="JobSheet" or "SeparatorSheet". The sheet is produced at the beginning of the job (for "JobSheet"), or at the beginning of each copy of each instance document (for "SeparatorSheet"), or is appended before the current <i>Component</i> . Contents for the sheet SHALL be drawn from the <i>StripMark</i> elements.
Interleaved	Valid for <i>@SheetType</i> ="SeparatorSheet". The sheet is produced after each page (e.g., used to insert sheets under transparencies). Contents for the sheet SHALL be drawn from the <i>StripMark</i> elements.

Table 6.175: SheetUsage Attribute Values (Sheet 2 of 2)

VALUE	DESCRIPTION
InterleavedBefore	Valid for @SheetType="SeparatorSheet". The sheet is produced before each page (e.g., used to insert sheets before transparencies). Contents for the sheet SHALL be drawn from the StripMark elements.
OnError	Valid for @SheetType="ErrorSheet". The sheet is produced at the end of the job only when an error or warning occurs.
Slip	Valid for @SheetType="SeparatorSheet". The sheet is produced between each copy of each instance document. Contents for the sheet SHALL be drawn from the StripMark elements.
SlipCopy	Valid for @SheetType="SeparatorSheet". 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 SHALL be drawn from the StripMark elements.
Trailer	Valid for @SheetType="AccountingSheet", "ErrorSheet", "JobSheet" and "SeparatorSheet". The sheet is produced at the end of the job (for "AccountingSheet", "ErrorSheet" and "JobSheet"), or at the end of each copy of each instance document (for "SeparatorSheet"), or is appended after the current sheet, signature, layout or RunList as defined by its context. Contents for the sheet SHALL be drawn from the StripMark elements. Note: Use @SheetType="ErrorSheet" and @SheetUsage="Trailer" to always produce a sheet that contains error or success information even if no errors or warnings occurred.

6.83 StitchingParams

StitchingParams provides the parameters for the Stitching process. The process coordinate system is defined as follows:

- The X-axis is aligned with the second registered edge, and it increases from the binding edge to the face edge.
- The Y-axis is aligned with the spine and increases from the first registered edge to the edge opposite to the registered face edge.

Note: If no spine exists, e.g. in side or corner stitching, the Y-axis and default reference edge is the left side of the sheet.

Note: The stitches are applied from the front in the figures describing the stitching coordinate system.

Resource Properties

Intent Pairing: **BindingIntent**

Input of Processes: **Stitching**

Table 6.176: StitchingParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
Angle ?	float	Angle of stitch in degrees. The angle increases in a counterclockwise direction. Horizontal="0", which means that it is parallel to the X-axis of the operation coordinate system. @Angle defaults to the system-specified value that MAY vary depending on other attributes set in this resource. If @StitchType="Saddle", @Angle SHALL NOT be specified.
NumberOfStitches ?	integer	@NumberOfStitches specifies the number of stitches.
Offset ?	float	Distance between stitch and binding edge. If @StitchType="Saddle", @Offset SHALL NOT be specified.
StapleShape ?	enumeration	Specifies the shape of the staples to be used. Allowed value is from: ▶ StapleShape.
StitchOrigin ?	enumeration	Defines the origin of @StitchPositions. For an illustration of the values, see ▶ Figure 6-40: Stitching coordinate system for @StitchOrigin values. Allowed values are: TrimBoxCenter TrimBoxJogSide UntrimmedJogSide
StitchPositions ?	FloatList	Array containing the stitch positions. The center of the stitch SHALL be specified, and the number of entries SHALL match the number given in @NumberOfStitches.

Table 6.176: StitchingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>StitchType</i> ?	enumeration	Specifies the type of the Stitching operation. Allowed values are: Corner – Stitch in the corner that is at the clockwise end of the reference edge. For example, to stitch in the upper right corner set Resource/@Orientation="Rotate90" . Saddle – Stitch on the middle fold, which is on the saddle. Side – Stitch along the reference edge. For example, to stitch in the top, set Resource/@Orientation="Rotate90" .
<i>StitchWidth</i> ?	float	Width of the stitch to be used.
<i>TightBacking</i> ?	enumeration	Definition of the geometry of the back of the product. See BlockPreparationParams/@TightBacking and ▶ Figure 6-2: Backing and Rounding measurements for TightBacking for details. Allowed value is from: ▶ TightBacking.
<i>WireGauge</i> ?	float	Gauge of the wire to be used.
<i>FileSpec (CIP3)</i> ?	element	Reference to a CIP3 file that contains stitching instructions in the ▶ [CIP3 - PPF] format.

Figure 6-38: Parameters and coordinate system used for saddle stitching

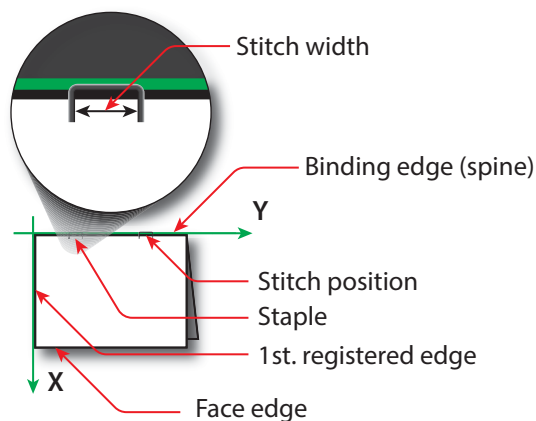


Figure 6-39: Parameters and coordinate system used for stitching

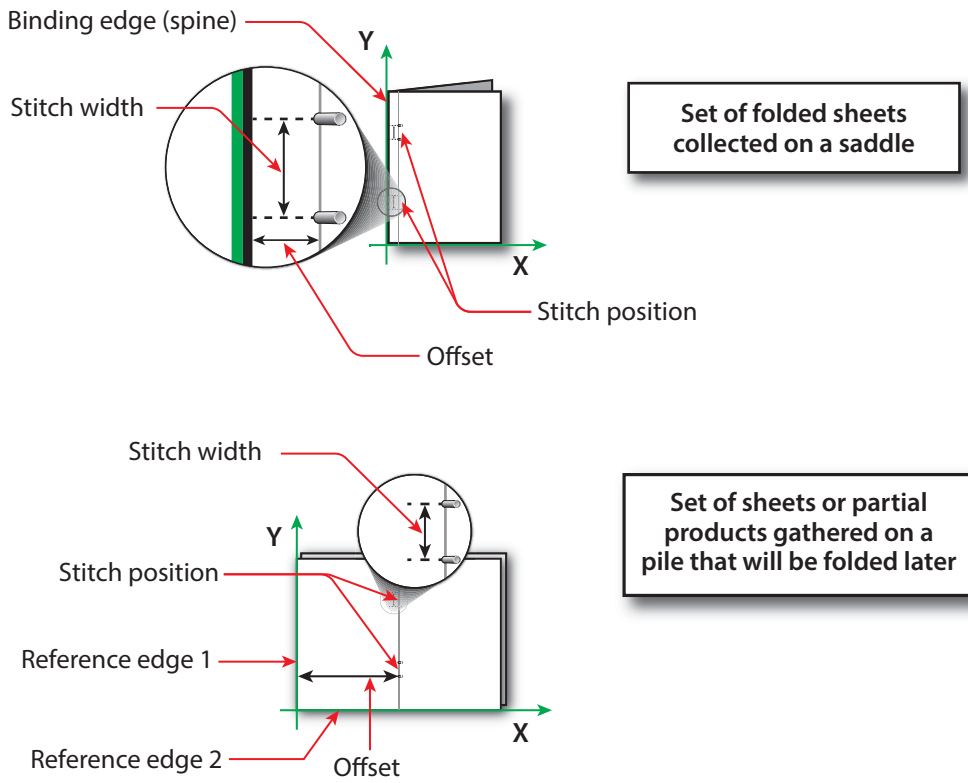
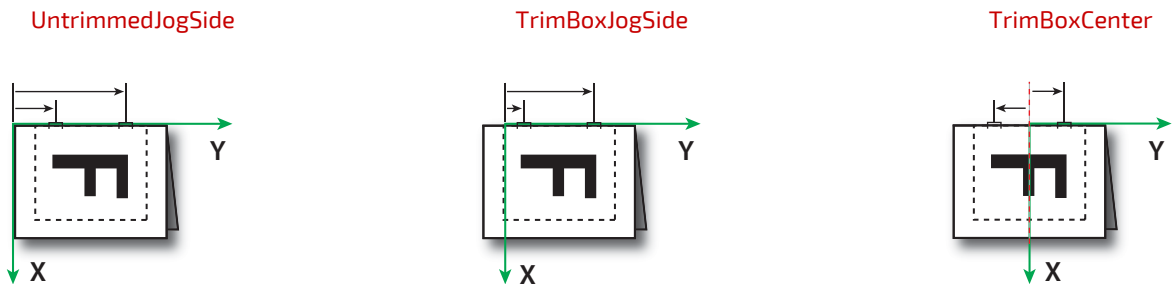


Figure 6-40: Stitching coordinate system for @StitchOrigin values



6.84 StrappingParams

StrappingParams defines the details of **Strapping**.

Resource Properties

Input of Processes: **Strapping**

Table 6.177: StrappingParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>StrappingType</i>	enumeration	Strapping pattern. Allowed values are: Cross – Two crossed straps. Double – Two parallel single straps. DoubleCross – Two cross straps that strap each side of a box. Single – One strap.

Table 6.177: StrappingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>StrapPositions</i> ?	FloatList	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 Y ₀ 0 0 Y ₁ 0" (see ▶ Figure 6-41: Strapped bundle and ▶ Figure 6-42: Strapped bundle with sub-bundles). A centered cross strap in the x-y plane would be specified as "x/2 0 0 0 y/2 0", which specifies one strap in the x-plane and another in the y-plane.

Figure 6-41: Strapped bundle

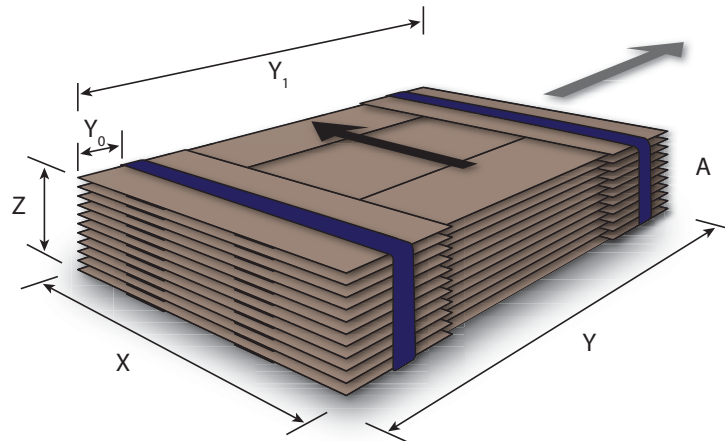
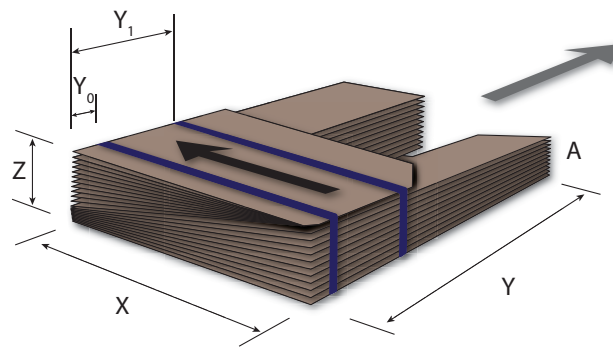


Figure 6-42: Strapped bundle with sub-bundles



6.85 ThreadSealingParams

ThreadSealingParams provides the parameters for the **ThreadSealing** process.

Resource Properties

Intent Pairing: **BindingIntent**

Input of Processes: **ThreadSealing**

Table 6.178: 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.
<i>ThreadLength</i> ?	float	Length of one thread.

Table 6.178: ThreadSealingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ThreadPositions</i> ?	FloatList	Array containing the y-coordinate of the center positions of the thread.
<i>ThreadStitchWidth</i> ?	float	Width of one stitch.

6.86 ThreadSewingParams

ThreadSewingParams provides the parameters for the **ThreadSewing** 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).

Resource Properties

Intent Pairing: **BindingIntent**

Input of Processes: **ThreadSewing**

Table 6.179: ThreadSewingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>BlindStitch</i> ?	boolean	A value of "true" specifies a blind stitch after the last stitch.
<i>NeedlePositions</i> ?	FloatList	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> ?	float	Specifies the distance between the stitch and the binding edge. Used only when <i>@SewingPattern</i> ="Side".
<i>SewingPattern</i> ?	enumeration	Sewing pattern. Allowed values are: CombinedStaggered Normal Side – Side sewing. Staggered
<i>ThreadThickness</i> ?	float	Thread thickness.

Figure 6-43: Parameters and coordinate system used for thread sewing

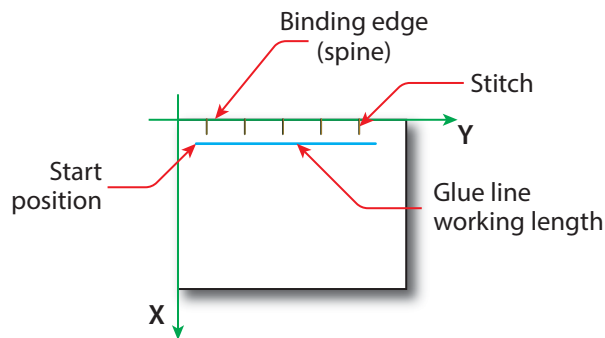
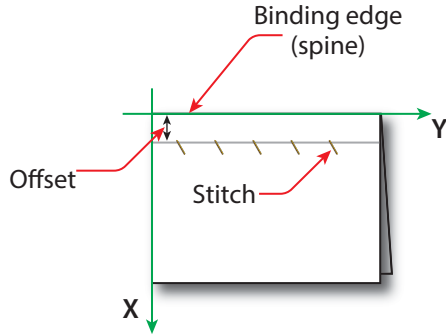


Figure 6-44: Parameters and coordinate system used for side sewing



6.87 Tool

A **Tool** defines a generic tool that can be customized for a given job (e.g., an embossing stamp) or an auxiliary device such as a fork lift. See also **Device** for description of the primary device that executes a process. The manufacturing process for the tool is not described within **XJDF**.

Resource Properties

Resource referenced by: **EmbossingParams/Emboss**

Input of Processes: **Any Process, Embossing, ShapeCutting**

Output of Processes: **DieMaking**

Table 6.180: Tool Resource

NAME	DATA TYPE	DESCRIPTION
<i>ToolType</i> ?	NMTOKEN	Type of the tool. Values include: Braille – Embossing tool for blind script. CentralStripper – The center tool of the stripper tool set. Stripping means removing small parts of waste in between blanks. ChangingCuttingBlock – 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 its own <i>@ExternalID</i> . CounterDie – The lower tool of the die-cut pair with the counter (female) parts for the creases. CutDie – The upper tool of the die-cut pair with the actual cutting and creasing knives. EmbossingCalendar EmbossingStamp ForkLift – A device used to lift forks. FrontWasteSeparator – The tool to remove gripper margin from the sheet. LowerBlanker – The lower tool of the blanker pair (blanking means separating blanks). LowerStripper – The lower tool of the stripper toolset. RollStand – A roll stand is a storage tool for Components . Components are rolled onto a roll stand with the Winding process. ToolSet – The value "ToolSet" is used when the <i>@ExternalID</i> refers not to a single tool, but to a set of matching tools that are used in the process (e.g., when <i>@ExternalID</i> is a single stock item number in the MIS for a tool set consisting of a "CutDie" and a "CounterDie"). UpperBlanker – The upper tool of the blanker pair. UpperStripper – The upper tool of the stripper toolset.
<i>IdentificationField</i> *	element	IdentificationField associates bar codes or labels with this Tool .

Figure 6-45: Roll stand



6.88 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: ▶ [CIP3 - PPF].

Resource Properties

Input of Processes: **Any Process**

Table 6.181: TransferCurve Resource

NAME	DATA TYPE	DESCRIPTION
<i>CTM</i> ?	matrix	@ <i>CTM</i> SHALL define the transformation of the coordinate system in the device, relative to the Layout coordinate system as defined by Part / @ <i>TransferCurveName</i> . Note: In JDF 1.x, CTM is applied relative to the prior coordinate system in the list of @ <i>Name</i> values whereas in XJDF CTM always applies relative to the Layout coordinate system.
<i>Curve</i> ?	Transfer-Function	The density mapping curve for this TransferCurve .

6.89 TrappingParams

TrappingParams provides a set of controls that are used to generate traps that are used to avoid mis-registration. These parameters MAY vary for different colorants so that **TrappingParams** MAY be partitioned by **Part**/**@Separation**.

Resource Properties

Input of Processes: **Trapping**

Table 6.182: TrappingParams Resource (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<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 image 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>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 6.182: TrappingParams Resource (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
MinimumBlackWidth ?	float	Specifies the minimum width, in points, of a trap that uses black ink. Allowable values are those greater than or equal to zero.
StepLimit ?	float	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 to or greater than the larger of 0.05 or @StepLimit times the lower value, i.e. if $\text{Value}_{hi} \geq \text{Value}_{lo} + \max(@\text{StepLimit} * \text{Value}_{lo}, 0.05)$ then the edge is a candidate for trapping. The value 0.05 is set to avoid trapping light areas in vignettes.
TrapColorScaling ?	float	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.
TrapWidth ?	XYPair	Specifies the trap width in the X and Y directions of the page or surface ByteMap .

6.90 TrimmingParams

TrimmingParams provides the parameters for the **Trimming** process.

Resource Properties

Input of Processes: **Trimming**

Table 6.183: TrimmingParams Resource

NAME	DATA TYPE	DESCRIPTION
Height ?	float	Height of the trimmed product.
TrimCover ?	enumeration	Specifies the covers to be trimmed. Covers containing flaps are generally not trimmed. Allowed values are: Back – Trim back cover only. Both – Trim front and back cover. Front – Trim front cover only. Neither – Do not trim cover.
TrimmingOffset ?	float	Amount to be cut from the bottom side.
Width ?	float	Width of the trimmed product.

6.91 UsageCounter

Many devices use counters to track equipment utilization or work performed, such as impressions produced or variable data documents generated. **UsageCounter** represents a type of equipment or software usage that is tracked by the value of a usage counter used by a device to record the amount of work performed. See ▶ Section 6.1.2 PartAmount. Default units are “countable objects”. See ▶ Section 1.9.1 Units of measurement.

Resource Properties

Input of Processes: **Any Process**

Table 6.184: UsageCounter Resource

NAME	DATA TYPE	DESCRIPTION
<i>CounterTypes</i> ?	NMTOKENS	This attribute indicates the types of usage being counted by the UsageCounter . Values include: Insert – Post fuser inserter. OneSided – Includes one sided counts. TwoSided – Includes two sided counts. NormalSize – Includes normal size counts. LargeSize – Includes large size counts. Black – Includes black colorant only counts. Color – Includes one or more non-black, non-highlight color colorants counts. Blank – Includes entirely blank counts. HighlightColor – Includes highlight colorant counts.
<i>Scope</i>	enumeration	The scope of this usage counter. Allowed values are: Job – Count in the context of one XJDF . Lifetime – Count since machine last had a firmware reset. SHALL NOT be specified when UsageCounter is used as a resource in an XJDF ticket. PowerOn – Count since the machine was powered on. SHALL NOT be specified when UsageCounter is used as a resource in an XJDF ticket.

6.92 VarnishingParams

VarnishingParams provides the parameters of a **Varnishing** process.

Resource Properties

Intent Pairing: **ColorIntent**

Input of Processes: **Varnishing**

Table 6.185: VarnishingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>ModuleID</i> ?	NMTOKEN	Identifier of the varnishing module in a multi-function device, such as a printing press. See Module for details.
<i>ModuleType</i> ?	enumeration	The type of module used to apply the varnish. Allowed values are: CoatingModule – The varnish is applied in a specialized coating unit. PrintModule – The varnish is applied in a printing unit.
<i>VarnishArea</i> ?	enumeration	Area to be varnished. @ VarnishArea specifies the requirements for ExposedMedia . Allowed values are: Full – The entire media surface SHALL be varnished. Spot – Only parts of the media surface SHALL be varnished.
<i>VarnishMethod</i> ?	enumeration	Method used for varnishing. Allowed values are: Blanket – Varnishing is performed in a dedicated coating module. An ExposedMedia that references a Media/@MediaType="Blanket" MAY be specified. Independent – No additional ExposedMedia is required. This method MAY be used to specify varnishing in a digital press. Plate – Varnishing is performed in a print module or a dedicated coating module. An ExposedMedia that references a Media/@MediaType="Plate" SHOULD be specified.

6.93 VerificationParams

VerificationParams provides the parameters of a **Verification** process.

Resource Properties

Input of Processes: **Verification**

Table 6.186: *VerificationParams Resource*

NAME	DATA TYPE	DESCRIPTION
<i>Tolerance</i> ?	float	Ratio of tolerated verification failures to the total number of tests. "0.0" = no failures allowed, "1.0" = all tests MAY fail.
<i>FileSpec</i> ?	element	Reference to data that contains implementation specific descriptions of the resources to be verified.

6.94 VerificationResult

VerificationResult defines the set of results from the **Verification** process.

Output of Processes: **Verification**

Table 6.187: *VerificationResult Resource*

NAME	DATA TYPE	DESCRIPTION
<i>Accepted</i> ?	integer	Number of resources that were correctly verified.
<i>Rejected</i> ?	integer	Number of resources that were not correctly verified.
<i>Unknown</i> ?	integer	Number of resources that were scanned but are not in the explicit or implied list of known resources.
<i>FileSpec (Accepted)</i> ?	element	Reference to data that contains implementation specific descriptions of the resources that were correctly verified.
<i>FileSpec (Rejected)</i> ?	element	Reference to data that contains implementation specific descriptions of the resources that were NOT correctly verified.
<i>FileSpec (Unknown)</i> ?	element	Reference to data that contains implementation specific descriptions of the resources that were scanned but are NOT in the explicit list of known resources.

6.95 WebInlineFinishingParams

WebInlineFinishingParams specifies the parameters for web inline finishing equipment using the **WebInlineFinishing** process.

Resource Properties

Input of Processes: **WebInlineFinishing**

Table 6.188: *WebInlineFinishingParams Resource*

NAME	DATA TYPE	DESCRIPTION
<i>FolderProduction</i> *	element	Specifies the folder setup for newspaper presses.
<i>ProductionPath</i> ?	element	<i>ProductionPath</i> describes the paper path that is used through the press and describes exactly one particular product that has to be produced.

RESOURCES

6.95.1 FolderProduction

Table 6.189: FolderProduction Element

NAME	DATA TYPE	DESCRIPTION
<i>ModuleID</i> ?	NMTOKEN	Identifies a particular folder module to be used. @ <i>ModuleID</i> SHALL match <i>Device/Module/@ModuleID</i> .
<i>ProductionType</i> ?	enumeration	Indicates whether the product is collected or not. Allowed values are: Collect NonCollect

6.95.2 ProductionPath

Table 6.190: 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.

6.96 WindingParams

The parameters for the **Winding** process.

Resource Properties

Input of Processes: **Winding**

Table 6.191: WindingParams Resource

NAME	DATA TYPE	DESCRIPTION
<i>Copies</i> ?	integer	Number of copies in one column that SHOULD be placed on a finished roll. At most, one of @ <i>Copies</i> , @ <i>Diameter</i> or @ <i>Length</i> SHOULD be specified.
<i>Diameter</i> ?	float	Outer diameter in points of the finished roll. At most one of @ <i>Copies</i> , @ <i>Diameter</i> or @ <i>Length</i> SHOULD be specified.
<i>Fixation</i> ?	NMTOKEN	Method specifying how the Component is attached to the core. Values include: DoubleSidedTape – Tape with adhesive on both sides. Glue Label – One of the output Component resources (self-adhesive labels) is used. None – No fixation is used. SingleSidedTape – Tape with adhesive on one side.
<i>Length</i> ?	float	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.

6.97 WrappingParams

WrappingParams defines the details of **Wrapping**. Details of the material used for **Wrapping** can be found in either the **MiscConsumable(Wrapper)** or **Component(Wrapper)** resources that are inputs of the **Wrapping** process.

Resource Properties

Input of Processes: **Wrapping**Table 6.192: *WrappingParams* Resource

NAME	DATA TYPE	DESCRIPTION
<i>WrappingKind</i>	enumeration	<p><i>@WrappingKind</i> specifies the wrapping method.</p> <p>Allowed values are:</p> <p>Band – The components are wrapped with a band. The material of the band is typically paper, plastic or rubber.</p> <p>LooseWrap – The wrap is loose around the component.</p> <p>ShrinkWrap – The wrap is shrunk around the component.</p>

7 Messaging

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 bootstrapping and setup
- Dynamic status, resource usage and error tracking for jobs and devices
- Pipe control
- Device setup and job changes
- Queue handling and job submission

This chapter specifies the XML structure of **XJMF**. For details of the exchange protocol and data packaging, see ▶ Section 9.5 XJDF and XJMF Interchange Protocol and ▶ Section 9.7 XJDF Packaging.

7.1 XJMF

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.

Table 7.1: XJMF Element

NAME	DATA TYPE	DESCRIPTION
<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>Header</i>	element	<i>Header</i> SHALL provide information about the sender of the XJMF package. If the sender is a proxy controller that forwards information from multiple devices, <i>XJMF/Header</i> SHALL provide information about the proxy controller. See also <i>Message/Header</i> .
<message elements> *	element	One or more messages SHALL be provided. The messages SHOULD be one of the message element types that are defined in XJMF , see ▶ Section 7.3 List of All XJMF Messages. The recipient SHALL process the messages in XML order.
<foreign namespace message elements> *	element	Any message elements in a foreign namespace. Foreign namespace extensions SHOULD NOT duplicate functionality of existing XJDF messages. They SHALL adhere to the structure that is defined in ▶ Section 7.1.1 Message and SHALL adhere to whichever of the message family definitions, as described in ▶ Section 7.2 XJMF Message Families, is appropriate. These elements MAY occur interleaved between messages in the XJDF namespace.

7.1.1 Message

The following table describes the contents of a message. A message is an abstract data type. ▶ Section 7.3 List of All XJMF Messages provides a list of all message element instances.

Table 7.2: Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	<i>Header</i> SHALL provide information about the original sending device of the individual message. <i>Header</i> SHALL be the first element in a <i>Message</i> , regardless of the alphabetical ordering of any other elements. The information in <i>Header</i> SHALL NOT be modified if the message is passed through a proxy controller. See also <i>XJMF/Header</i> .

7.1.2 Header

Header SHALL provide information about the sender of an audit, message or **XJMF**. The information in *Message/Header* and *AuditXXX/Header* SHALL NOT be modified and SHALL represent the status of the original message or audit if the message is passed through a proxy controller. The information in *XJMF/Header* SHALL be modified to represent the status of the proxy controller if the message is passed through a proxy controller.

The *Header* element SHALL be the first element in any audit or message as shown in those tables.

Table 7.3: Header

NAME	DATA TYPE	DESCRIPTION
<i>AgentName</i> ?	string	The name of the application that generated the parent message or 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 parent message or XJMF . The format of the version string MAY vary from one application to another, but SHOULD be consistent for an individual application.
<i>Author</i> ?	string	Human readable description of the employee that entered the message. <i>XJMF/Header/@Author</i> SHOULD NOT be specified.
<i>DeviceID</i>	NMTOKEN	Unique identifier of the sender.
<i>ICSVersions</i> ?	NMTOKENS	CIP4 Interoperability Conformance Specification (ICS) Versions that the sender complies with. The value of <i>@ICSVersions</i> SHALL conform to the value format described in ▶ Section 3.1.1 ICS Versions Value.
<i>ID</i> ?	ID	Identifies the parent message. <i>@ID</i> SHALL be present if <i>Subscription</i> is present in the parent <i>Query</i> .
<i>PersonalID</i> ?	NMTOKEN	Machine readable identifier of the employee that entered the message. <i>XJMF/Header/@PersonalID</i> SHOULD NOT be specified.
<i>refID</i> ?	NMTOKEN	<i>@refID</i> SHALL identify a message that the parent of this <i>Header</i> responds to. If the parent is a <i>Response</i> , this SHALL be <i>Header/@ID</i> of the initiating <i>Query</i> or <i>Command</i> . If the parent is a <i>Signal</i> , this SHALL be <i>Header/@ID</i> of the initiating <i>Query</i> . Note: The data type is NMTOKEN because the referenced <i>Header/@ID</i> need not be in the same XML document.
<i>Time</i>	dateTime	Date and time when the message was generated.

7.2 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. Message families are abstract data types, ▶ Section 7.3 List of All XJMF Messages provides a list of all message family element instances.

7.2.1 Query

A *Query* is an abstract message that retrieves information from a receiver without changing the state of that receiver. For details of **XJMF** handshaking, see ▶ Section 9.6 XJMF Handshaking. The following table shows the content of a *Query* message.

Table 7.4: Query

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See <i>Message/Header</i> .
<i>Subscription</i> ?	element	A <i>Subscription</i> is a request for a persistent channel. If present, <i>Subscription</i> SHALL be specified after <i>Header</i> and prior to any other elements in a <i>Query</i> . Any other elements in the <i>Query</i> will then be alphabetically ordered in the normal way. If <i>Subscription</i> is present then <i>Header/@ID</i> shall be specified. For details of creating and managing persistent channels see ▶ Section 9.6.3 Managing Persistent Channels

7.2.1.1 Subscription

A *Subscription* specifies the target URL for signals and optionally, additional details of the persistent channel. See ▶ Section 9.6.2 Subscribing for Signals for details.

Table 7.5: 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 <i>Response</i> SHOULD indicate <i>@ReturnCode</i> 111, which is “Subscription request denied”. Allowed value is from: ▶ <i>ChannelMode</i> . Note: See <i>Signal/@ChannelMode</i> .
<i>Languages</i> ?	languages	List of languages selected for human readable communication. If not specified, the operating system language SHOULD be used. If multiple languages are specified, the second and further languages SHOULD only be used for providing additional localized <i>Comment</i> elements. Messages SHALL NOT be sent multiple times for the same event.
<i>RepeatTime</i> ?	float	Requests an update <i>Signal</i> every <i>@RepeatTime</i> seconds. If specified, the <i>Signal</i> SHALL be generated periodically independent of any other trigger conditions. <i>@RepeatTime</i> SHALL NOT override any <i>Signals</i> triggered by a change of status. <i>Signals</i> triggered by a status change SHALL be sent regardless of the value of <i>@RepeatTime</i> . A sender MAY restart counting for <i>@RepeatTime</i> based <i>Signals</i> whenever it sends a <i>Signal</i> to the same subscription.
<i>URL</i>	URL	URL of the persistent channel receiving end. The protocol of the <i>Subscription</i> is specified by the scheme of <i>@URL</i> .

7.2.2 Command

A *Command* is syntactically equivalent to a *Query*, but rather than simply retrieving information, it is an abstract message that causes a state change in the receiver. For details of XJMF handshaking, see ▶ Section 9.6 XJMF Handshaking. The following table contains the contents of a *Command* message.

Table 7.6: Command Family

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See <i>Message/Header</i> .

7.2.3 Signal

A *Signal* is an abstract message element 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 9.6 XJMF Handshaking. For details of setting up subscriptions for signals, see ▶ Section 9.6.2 Subscribing for Signals. The following table shows the contents of a *Signal*.

Table 7.7: Signal Family

NAME	DATA TYPE	DESCRIPTION
<i>ChannelMode</i> ?	enumeration	Specifies reliability of the signal. Allowed value is from: ▶ <i>ChannelMode</i> .
<i>Header</i>	element	See <i>Message/Header</i> .

7.2.4 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 9.6 XJMF Handshaking. The following table shows the content of a **Response** message.

Table 7.8: Response Family

NAME	DATA TYPE	DESCRIPTION
<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 B Return Values. Note: Additional values MAY be specified in ICS documents.
<i>Header</i>	element	See <i>Message/Header</i> .
<i>Notification</i> ?	element	Additional information including textual description of the return code. The <i>Notification</i> element SHOULD be provided if the <i>@ReturnCode</i> is greater than 0. See ▶ Section 7.10 Notification. If present then the <i>Notification</i> element SHALL be placed after <i>Header</i> and prior to any other elements in a Response message.

7.3 List of All XJMF Messages

The following table provides a list of all message element types.

Table 7.9: List of XJMF Messages (Sheet 1 of 2)

MESSAGES	DESCRIPTION
<i>CommandForceGang</i> <i>ResponseForceGang</i>	A gang is forced to execute.
<i>QueryGangStatus</i> <i>ResponseGangStatus</i> <i>SignalGangStatus</i>	The status of a gang is queried.
<i>QueryKnownDevices</i> <i>ResponseKnownDevices</i> <i>SignalKnownDevices</i>	Returns information about the devices that are controlled by a controller.
<i>QueryKnownMessages</i> <i>ResponseKnownMessages</i>	Returns a list of all messages that are supported by the controller.
<i>QueryKnownSubscriptions</i> <i>ResponseKnownSubscriptions</i> <i>SignalKnownSubscriptions</i>	Returns a list of active persistent channels.
<i>CommandModifyQueueEntry</i> <i>ResponseModifyQueueEntry</i>	Modifies the properties of one or more <i>QueueEntry</i> elements.
<i>QueryNotification</i> <i>ResponseNotification</i> <i>SignalNotification</i>	Used to signal events due to any activities of a device, operator, etc. Generally sent as signals. <i>QueryNotification</i> allows subscriptions for <i>SignalNotification</i> messages.

Table 7.9: List of XJMF Messages (Sheet 2 of 2)

MESSAGES	DESCRIPTION
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 ResponseRequestQueueEntry	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. This message can also be used to query the level of consumables in a device.
CommandResubmitQueueEntry ResponseResubmitQueueEntry	Replaces a queue entry without affecting the entry's parameters. CommandResubmitQueueEntry is used, for example, for late changes to a submitted XJDF .
CommandReturnQueueEntry ResponseReturnQueueEntry	Returns a job that had been submitted with a SubmitQueueEntry to the controller that originally submitted the job.
CommandShutDown ResponseShutDown	Shuts down a device.
QueryStatus ResponseStatus SignalStatus	Queries or signals the general status of a device, controller or job.
CommandStopPersistentChannel ResponseStopPersistentChannel	Closes a persistent channel.
CommandSubmitQueueEntry ResponseSubmitQueueEntry	Submits an XJDF to a queue in order to be executed.
CommandWakeUp ResponseWakeUp	Wakes up a device that is in standby mode.

7.4 ForceGang

The [ForceGang](#) message forces all the selected [QueueEntry](#)[@Status="Waiting"] elements that belong to a gang 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.

7.4.1 CommandForceGang

Table 7.10: CommandForceGang Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
GangCmdFilter	element	Defines the gang(s) to be forcibly executed.

7.4.1.1 GangCmdFilter

Table 7.11: GangCmdFilter Element

OBJECT TYPE	ELEMENT NAME	DESCRIPTION
GangNames ?	NMTOKENS	A list of @GangName values of the gang(s) to be selected.

Table 7.11: GangCmdFilter Element

OBJECT TYPE	ELEMENT NAME	DESCRIPTION
<i>Policy</i> ?	enumeration	The policy with which the elements in the gang SHALL be processed. Allowed values are: All - All elements in a given gang SHALL be processed. Optimized - As many elements in a given gang as can be processed without unnecessary waste SHOULD be processed. The algorithm for selecting the respective elements is implementation dependent and SHOULD take priority and scheduling data into account.

7.4.2 ResponseForceGang

Table 7.12: ResponseForceGang Message

NAME	DATA TYPE	DESCRIPTION
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode .
<i>Header</i>	element	See Message/Header .
<i>Notification</i> ?	element	See Response/Notification .

7.5 GangStatus

[GangStatus](#) returns a description of the gang(s). Details are specified in the [GangInfo](#) element.

7.5.1 QueryGangStatus

Table 7.13: QueryGangStatus Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See Message/Header .
<i>Subscription</i> ?	element	See Query/Subscription .
<i>GangQuFilter</i> ?	element	Defines a filter for the gang(s) that are queried. If GangQuFilter is not supplied, all gangs are queried.

7.5.1.1 GangQuFilter

Table 7.14: GangQuFilter Element

NAME	DATA TYPE	DESCRIPTION
<i>GangNames</i> ?	NMTOKENS	@ GangName of the gang(s) being queried.

7.5.2 ResponseGangStatus

Table 7.15: ResponseGangStatus Message

NAME	DATA TYPE	DESCRIPTION
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode .
<i>Header</i>	element	See Message/Header .
<i>Notification</i> ?	element	See Response/Notification .
<i>GangInfo</i> *	element	Describes the status of the gang(s).

7.5.2.1 GangInfo

Details of the gang are specified in [GangInfo](#) elements.

Table 7.16: GangInfo Element

NAME	DATA TYPE	DESCRIPTION
<i>Amount</i> ?	float	Quantity of <i>QueueEntry</i> items that are currently waiting to be executed. If the device specifies amount in a unit other than countable objects, such as m ² , <i>@Amount</i> SHALL be specified in the units of the device.
<i>GangName</i>	NMTOKEN	Name of the gang.

7.5.3 SignalGangStatus

Table 7.17: SignalGangStatus Message

NAME	DATA TYPE	DESCRIPTION
<i>ChannelMode</i> ?	enumeration	Allowed value is from: ▶ ChannelMode. Note: See <i>Signal/@ChannelMode</i> .
<i>Header</i>	element	See <i>Message/Header</i> .
<i>GangInfo</i> *	element	Describes the status of the gang(s).

7.6 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 list of *Device* elements. ▶ Example 7.1:KnownDevices Response shows the response from a press controller that controls two physical presses.

Example 7.1: KnownDevices Response

```
<XJMF xmlns="http://www.CIP4.org/JDFSSchema_2_0">
  <Header DeviceID="VeggieController" ID="1_000002" Time="2018-02-28T16:00:19+00:00"/>
  <ResponseKnownDevices ReturnCode="0">
    <Header DeviceID="VeggieController" ID="R1"
      Time="2018-02-28T16:00:19+00:00" refID="Q1"/>
    <Device DeviceID="dev1" DeviceType="ACME Linda potato press V16-12" XJMFURL="http://acmepota-
tol:1234/xjmfurl"/>
    <Device DeviceID="dev2"
      DeviceType="ACME Baldrick turnip press V42-66" XJMFURL="http://acmeturnip1:1234/xjmfurl"/>
    <!-- One Device element for each known device follows here -->
  </ResponseKnownDevices>
</XJMF>
```

7.6.1 QueryKnownDevices

Table 7.18: QueryKnownDevices Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See <i>Message/Header</i> .
<i>Subscription</i> ?	element	See <i>Query/Subscription</i> .
<i>DeviceFilter</i> ?	element	<i>DeviceFilter</i> refines the level of detail of the list of devices queried.

7.6.1.1 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 7.19: DeviceFilter Element

NAME	DATA TYPE	DESCRIPTION
<i>DeviceDetails</i> ?	enumeration	<p><i>@DeviceDetails</i> refines the level of provided information about the device.</p> <p>Allowed values are:</p> <p>Brief – Provide only <i>Device/@DeviceID</i>.</p> <p>Full – Provide maximum available device information including all <i>Module</i> and <i>Device</i> descriptions.</p> <p>Modules – <i>Module</i> elements SHALL be provided if the device has modules.</p>

7.6.2 ResponseKnownDevices

Table 7.20: ResponseKnownDevices Message

NAME	DATA TYPE	DESCRIPTION
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode .
<i>Header</i>	element	See Message/Header .
<i>Notification</i> ?	element	See Response/Notification .
<i>Device</i> *	element	Each <i>Device</i> SHALL represent one of the known devices.

7.6.3 SignalKnownDevices

Table 7.21: SignalKnownDevices Message

NAME	DATA TYPE	DESCRIPTION
<i>ChannelMode</i> ?	enumeration	Allowed value is from: ▶ ChannelMode Note: See Signal/@ChannelMode .
<i>Header</i>	element	See Message/Header .
<i>Device</i> *	element	Each <i>Device</i> SHALL represent one of the known devices.

7.7 KnownMessages

The [KnownMessages](#) query message returns a list of all message types that are supported by the controller.

7.7.1 QueryKnownMessages

Table 7.22: QueryKnownMessages Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See Message/Header .

7.7.2 ResponseKnownMessages

Table 7.23: ResponseKnownMessages Message

NAME	DATA TYPE	DESCRIPTION
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode .
<i>Header</i>	element	See Message/Header .
<i>Notification</i> ?	element	See Response/Notification .
<i>MessageService</i> *	element	Each <i>MessageService</i> SHALL specify one supported message.

7.7.2.1 MessageService

The response is a list of *MessageService* elements. Each *MessageService* SHALL specify one explicit message type that is supported by the device.

Table 7.24: MessageService Element

NAME	DATA TYPE	DESCRIPTION
<i>ResponseModes</i> ?	enumerations	Specifies the supported synchronous <i>Response</i> or <i>Signal</i> channel modes with which the receiver can reply to a <i>Query</i> . <i>@ResponseModes</i> SHALL NOT be specified unless <i>@Type</i> specifies a <i>Query</i> message. Allowed values are: <i>FireAndForget</i> – The response to the message is implemented as a persistent channel <i>Signal</i> with <i>Signal/@ChannelMode="FireAndForget"</i> . <i>Reliable</i> – The response to the message is implemented as a persistent channel <i>Signal</i> with <i>Signal/@ChannelMode="Reliable"</i> . <i>Response</i> – The response to the message is implemented as a synchronous <i>Response</i> .
<i>Type</i>	NMTOKEN	Name of the supported message element, e.g. <i>QueryKnownMessages</i> .
<i>URLSchemes</i> ?	enumerations	List of schemes supported for the message defined by this <i>MessageService</i> . Allowed values are: <i>http</i> – HTTP (Hypertext Transport Protocol) <i>https</i> – HTTPS (Hypertext Transport Protocol – Secure)

Example 7.2: KnownMessages Response

The following is an example of a *ResponseKnownMessages*.

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="DeviceID" ID="1_000002" Time="2018-02-28T16:00:17+00:00"/>
  <ResponseKnownMessages ReturnCode="0">
    <Header DeviceID="DeviceID" ID="R1" Time="2018-02-28T16:00:17+00:00" refID="Q1"/>
    <MessageService ResponseModes="Response" Type="QueryKnownMessages"/>
    <MessageService ResponseModes="FireAndForget Reliable" Type="QueryStatus"/>
    <MessageService Type="CommandSubmitQueueEntry"/>
    <MessageService Type="ResponseReturnQueueEntry"/>
  </ResponseKnownMessages>
</XJMF>
```

7.8 KnownSubscriptions

The *KnownSubscriptions* message enables controllers to query devices for a list of active persistent channels.

7.8.1 QueryKnownSubscriptions

Table 7.25: QueryKnownSubscriptions Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
Subscription ?	element	See Query/Subscription .
SubscriptionFilter ?	element	This SubscriptionFilter selects which subscriptions SHALL be included in the returned messages' list of SubscriptionInfo elements. If not specified a SubscriptionInfo element is supplied for all known subscriptions.

7.8.1.1 SubscriptionFilter

The [SubscriptionFilter](#) element is a filter to limit the list of [SubscriptionInfo](#) elements that are returned in the [KnownSubscriptions](#) response.

Table 7.26: SubscriptionFilter Element

NAME	DATA TYPE	DESCRIPTION
DeviceID ?	NMTOKEN	Only SubscriptionInfo elements for subscriptions from devices or controllers with a matching @DeviceID attribute SHALL be returned.
URL ?	URL	URL of the receiving controller. This SHALL be identical to the Subscription/@URL that was used to create the persistent channel. Only SubscriptionInfo elements with a matching value of SubscriptionInfo/@URL SHALL be returned.

7.8.2 ResponseKnownSubscriptions

Table 7.27: ResponseKnownSubscriptions Message

NAME	DATA TYPE	DESCRIPTION
ReturnCode ?	integer	See Response/@ReturnCode .
Header	element	See Message/Header .
Notification ?	element	See Response/Notification .
SubscriptionInfo *	element	One SubscriptionInfo SHALL be provided for each active persistent channel that is selected by QueryKnownSubscriptions/SubscriptionFilter .

7.8.3 SignalKnownSubscriptions

Table 7.28: SignalKnownSubscriptions Message

NAME	DATA TYPE	DESCRIPTION
ChannelMode ?	enumeration	Allowed value is from: ▶ ChannelMode. Note: See Signal/@ChannelMode .
Header	element	See Message/Header .
SubscriptionInfo *	element	One SubscriptionInfo SHALL be provided for each active persistent channel that is selected by QueryKnownSubscriptions/SubscriptionFilter .

7.9 ModifyQueueEntry

[ModifyQueueEntry](#) modifies the state or position of one of more [QueueEntry](#) elements that are selected by [QueueFilter](#). [@Operation](#) specifies the operation that SHALL be applied to the selected queue entries.

See also [ResubmitQueueEntry](#) command for modifications of the underlying **XJDF** without modifying the queues.

7.9.1 CommandModifyQueueEntry

Table 7.29: CommandModifyQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See <i>Message/Header</i> .
<i>ModifyQueueEntryParams</i>	element	<i>ModifyQueueEntryParams</i> defines the selected queue entries and the operation to be performed.

7.9.1.1 ModifyQueueEntryParams

Table 7.30: ModifyQueueEntryParams Element

NAME	DATA TYPE	DESCRIPTION
<i>GangName</i> ?	NMTOKEN	Name of the gang that all <i>QueueEntry</i> items selected by the <i>QueueFilter</i> SHALL be moved to. <i>@GangName</i> SHALL NOT be specified unless <i>@Operation="SetGang"</i> . If <i>@Operation="SetGang"</i> and <i>@GangName</i> is not specified, then all selected <i>QueueEntry</i> items SHALL be removed from their current gang.
<i>NextQueueEntryID</i> ?	NMTOKEN	<i>QueueEntry/@QueueEntryID</i> of the queue entry that SHALL be positioned directly behind the queue entries that are selected by <i>QueueFilter</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>@NextQueueEntryID</i> SHALL NOT be specified unless <i>@Operation="Move"</i> .
<i>Operation</i>	enumeration	The operation that SHALL be performed on the queue entries that are selected by <i>QueueFilter</i> . Allowed value is from: ▶ Table 7.31 Operation Attribute Values.
<i>Position</i> ?	integer	Position in the queue where the queue entries that are selected by <i>QueueFilter</i> SHALL be moved to. Note: 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>@Position</i> SHALL NOT be specified unless <i>@Operation="Move"</i> .
<i>PrevQueueEntryID</i> ?	NMTOKEN	ID of the queue entry that SHALL be positioned directly in front of the queue entries that are selected by <i>QueueFilter</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> SHALL NOT be specified unless <i>@Operation="Move"</i> .
<i>Priority</i> ?	integer	New priority of the <i>QueueEntry</i> elements that are selected by <i>QueueFilter</i> . Priority is a number from 0 to 100, where "0" is the lowest priority and "100" is the 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>@Priority</i> SHALL NOT be specified unless <i>@Operation="Move"</i> .
<i>QueueFilter</i>	element	This <i>QueueFilter</i> selects the <i>QueueEntry</i> elements that the operation SHALL be applied to.

Table 7.31: Operation Attribute Values

VALUE	DESCRIPTION
Abort	The QueueEntry elements selected by QueueFilter SHALL be aborted and remain in the Queue with QueueEntry/@Status="Aborted" . ProcessRun/@EndStatus and NodeInfo/@Status of the XJDF that represents the queue entry SHALL be set to "Aborted" and the XJDF SHALL be delivered to the URL as specified by QueueSubmissionParams/@ReturnJMF .
Complete	The QueueEntry elements selected by QueueFilter SHALL be stopped and remain in the Queue with QueueEntry/@Status="Completed" . ProcessRun/@EndStatus and NodeInfo/@Status of the XJDF that represents the queue entry SHALL be set to "Completed" and the XJDF SHALL be delivered to the URL as specified by QueueSubmissionParams/@ReturnJMF .
Hold	If QueueEntry/@Status is "Waiting", QueueEntry/@Activation SHALL be set to "Held". The "Hold" operation SHALL NOT be applied to QueueEntry elements with a @Status other than "Waiting" or an @Activation other than "Active". If QueueEntry/@GangPolicy is other than "NoGang", a held QueueEntry retains its respective gang data but SHALL NOT influence execution of other QueueEntry elements that are in the gang.
Move	The position of the QueueEntry elements selected by QueueFilter SHALL be modified. The position of a QueueEntry SHALL NOT be modified unless @Status="Waiting" or @Status="Held" . If ModifyQueueEntryParams/@Priority is not specified, then each QueueEntry element selected by ModifyQueueEntryParams/QueueFilter SHALL have @Priority set to a value based upon its new Queue position such that it is smaller than or equal to the value of @Priority of the QueueEntry than precedes it and is greater than or equal to the value of @Priority of the QueueEntry that follows it. Note: The requirement to set the value of @Priority ensures that the QueueEntry elements in a Queue are always sorted by @Priority .
Remove	The QueueEntry elements selected by QueueFilter SHALL be removed from the queue. The Remove operation SHALL NOT be applied if QueueEntry [@Status="InProgress" or @Status="Suspended"].
Resume	The QueueEntry elements selected by QueueFilter SHALL be resumed. If QueueEntry/@Activation="Held" , QueueEntry/@Activation SHALL be set to "Active". If QueueEntry/@Status="Suspended" , QueueEntry/@Status SHALL be set to "InProgress". If QueueEntry/@GangPolicy is other than "NoGang", a resumed QueueEntry joins its respective gang.
SetGang	QueueEntry/@GangName of the items selected by QueueFilter SHALL be set to the value of @GangName . The "SetGang" operation SHALL NOT be applied unless QueueEntry/@Status="Waiting" .
Suspend	The QueueEntry elements selected by QueueFilter SHALL be suspended and its @Status set to "Suspended" if its @Status is "InProgress". Whether other queue entries can be run while the queue entries remain suspended depends on implementation. The "Suspend" operation has no effect on QueueEntry elements with a @Status other than "InProgress".

7.9.2 ResponseModifyQueueEntry

Table 7.32: ResponseModifyQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
ReturnCode ?	integer	See Response/@ReturnCode . In case of at least one failure due to an incorrect QueueEntry/@Status , @ReturnCode SHALL be set to a non-zero value that SHOULD be in the range of 105-116. See ▶ Table B.1 Return codes for XJMF for details.
Header	element	See Message/Header .
Notification ?	element	See Response/Notification .
QueueEntry *	element	Describes the selected QueueEntry elements after the command has been executed.

Example 7.3: Resuming a Queue Entry

The following examples illustrate the **Command** and **Response** sequence to resume a previously held **QueueEntry** with `@JobID="j1"`.

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:18+00:00"/>
  <CommandModifyQueueEntry>
    <Header DeviceID="TestSender" ID="C1" Time="2018-02-28T16:00:18+00:00"/>
    <ModifyQueueEntryParams Operation="Resume">
      <QueueFilter JobID="j1"/>
    </ModifyQueueEntryParams>
  </CommandModifyQueueEntry>
</XJMF>

<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="DeviceID" ID="1_000002" Time="2018-02-28T16:00:19+00:00"/>
  <ResponseModifyQueueEntry ReturnCode="0">
    <Header DeviceID="DeviceID" ID="R1" Time="2018-02-28T16:00:19+00:00" refID="C1"/>
    <QueueEntry Activation="Active" JobID="j1" QueueEntryID="QE1" Status="Waiting"/>
  </ResponseModifyQueueEntry>
</XJMF>
```

7.10 Notification

Notification messages are generally sent as **Signals**. **QueryNotification** 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).

7.10.1 QueryNotification

Table 7.33: QueryNotification Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
Subscription ?	element	See Query/Subscription .
NotificationFilter ?	element	Defines the types of Notification elements that SHALL be signaled.

7.10.1.1 NotificationFilter

Table 7.34: NotificationFilter Element

NAME	DATA TYPE	DESCRIPTION
Classes ?	enumerations	Defines the set of Notification/@Class types to be queried/subscribed for. If @Classes is not specified then all Notification classes are queried or subscribed to. Allowed values are from: ▶ Severity.
MilestoneTypes ?	NMTOKENS	Matching milestone types SHALL be returned and/or subscribed to. If @MilestoneTypes is not specified then all supported milestone values are queried or subscribed to. Values include those from: ▶ Milestones.

7.10.2 ResponseNotification

Table 7.35: ResponseNotification Message (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
ReturnCode ?	integer	See Response/@ReturnCode .

Table 7.35: ResponseNotification Message (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See Message/Header .
<i>Notification</i> ?	element	Notification that describes the event. See Notification , and also Response/Notification .

7.10.3 SignalNotification

Table 7.36: SignalNotification Message

NAME	DATA TYPE	DESCRIPTION
<i>ChannelMode</i> ?	enumeration	Allowed value is from: ▶ ChannelMode. Note: See Signal/@ChannelMode .
<i>Header</i>	element	See Message/Header .
<i>Notification</i>	element	Notification that describes the event. See ▶ Section 7.10 Notification.

7.11 PipeControl

[CommandPipeControl](#) modulates a flow of resources in a pipe. The type of pipe operation SHALL be specified in [PipeParams/@Operation](#). A pipe describes the consumption of a resource (by a consuming device) that commences when some lesser quantity of the resource becomes available without having to wait for the production device to complete production of the entire quantity.

See ▶ Section 9.3.5 Overlapping Processing for a more detailed discussion regarding the use of ▶ PipeControl messages.

7.11.1 CommandPipeControl

Table 7.37: CommandPipeControl Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See Message/Header .
<i>PipeParams</i>	element	Details of the PipeControl message.

7.11.1.1 PipeParams

Table 7.38: PipeParams Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>JobID</i>	NMTOKEN	Specifies XJDF/@JobID of the process at the receiving end.
<i>JobPartID</i> ?	NMTOKEN	Specifies XJDF/@JobPartID of the process at the receiving end.
<i>Operation</i>	enumeration	@Operation specifies whether the flow is being pushed or pulled. Allowed values are: Close – 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. Pause – 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. Pull – The PipeControl is a request to the producer to create resources by the consumer of the resources. Push – The PipeControl is a notification to the consumer that resources have been created by the producer of the resources.
<i>PipeID</i>	NMTOKEN	ResourceSet/Dependent/@PipeID at the receiving end. @PipeID SHALL be unique in the scope of the job that is selected by @JobID .

Table 7.38: PipeParams Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
MISDetails ?	element	Definition of how the costs for the production of the Resource are to be charged.
ResourceSet ?	element	Updated ResourceSet that SHALL be used by the process that receives the PipeControl command.

7.11.2 ResponsePipeControl

[ResponsePipeControl](#)/[@ReturnCode](#) SHALL be set to 0 if the [CommandPipeControl](#) has been accepted by the receiver. If not successful the [@ReturnCode](#) SHALL be set to one of the codes shown in ▶ Appendix B Return Values.

[@ReturnCode](#)="0" only specifies that the [CommandPipeControl](#) has been received and can be processed. Any problems that occur during processing of the resources and that lead to an interruption of the pipe SHALL be communicated with the appropriate [CommandPipeControl](#) messages.

Table 7.39: ResponsePipeControl Message

NAME	DATA TYPE	DESCRIPTION
ReturnCode ?	integer	See Response / @ReturnCode .
Header	element	See Message / Header .
Notification ?	element	See Response / Notification .

7.12 QueueStatus

[QueueStatus](#) returns a description of the current state of a [Queue](#).

7.12.1 QueryQueueStatus

Table 7.40: QueryQueueStatus Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message / Header .
Subscription ?	element	See Query / Subscription .
QueueStatusParams	element	QueueStatusParams defines a filter for the QueueStatus message.

7.12.1.1 QueueStatusParams

Table 7.41: QueueStatusParams Element

NAME	DATA TYPE	DESCRIPTION
UpdateGranularity ?	enumeration	Specifies whether all or only the updated QueueEntry elements should be included in the Queue . Allowed value is from: ▶ UpdateGranularity. Note: The first instance of a Signal shall result in the Queue describing all jobs.
QueueFilter ?	element	Filter that selects the QueueEntry elements that SHALL be returned in the Queue element of the response.

7.12.2 ResponseQueueStatus

Table 7.42: ResponseQueueStatus Message (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
ReturnCode ?	integer	See Response / @ReturnCode .
Header	element	See Message / Header .

Table 7.42: ResponseQueueStatus Message (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
Notification ?	element	See Response/Notification .
Queue ?	element	Describes the status of the queue.

7.12.3 SignalQueueStatus

Table 7.43: SignalQueueStatus Message

NAME	DATA TYPE	DESCRIPTION
ChannelMode ?	enumeration	Allowed value is from: ▶ ChannelMode. Note: See Signal/@ChannelMode .
Header	element	See Message/Header .
Queue	element	Describes the status of the queue.

7.12.4 Queue

Table 7.44: Queue Element

NAME	DATA TYPE	DESCRIPTION
MaxQueueSize ?	integer	The maximum number of QueueEntry elements excluding QueueEntry[@Status="Completed"] or QueueEntry[@Status="Aborted"] elements that can be contained in the Queue .
QueueSize ?	integer	The total number of QueueEntry elements that are in the Queue regardless of the settings in the QueueFilter . Thus the value of @QueueSize may be higher than the number of QueueEntry elements.
UpdateGranularity ?	enumeration	Specifies whether all or only the updated QueueEntry elements are included in the Queue . Allowed value is from: ▶ UpdateGranularity. Note: The first instance of a Signal shall result in the Queue describing all jobs.
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 . A QueueEntry is not automatically deleted when executed or aborted, but rather it remains in the Queue and its @Status is changed to "Completed" or "Aborted" accordingly.

7.13 RequestQueueEntry

This command requests a new queue entry from a potential submitting controller. The actual submission is still handled by [CommandSubmitQueueEntry](#).

Note: This command is emitted from the device that is represented 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.

7.13.1 CommandRequestQueueEntry

Table 7.45: CommandRequestQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
RequestQueueEntry Params	element	Defines the specifics for the requested job.

7.13.1.1 RequestQueueEntryParams

Table 7.46: RequestQueueEntryParams Element

NAME	DATA TYPE	DESCRIPTION
Activation ?	enumeration	Specifies the activation of the requested QueueEntry . Allowed value is from: ▶ Activation.
JobID ?	NMTOKEN	@ JobID of the requested QueueEntry .
JobPartID ?	NMTOKEN	@ JobPartID of the requested QueueEntry .
QueueURL	URL	URL of the Queue device that is requesting the QueueEntry and will accept Queue manipulation messages.
Part *	element	Partition parts of the requested QueueEntry .

7.13.2 ResponseRequestQueueEntry

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

Table 7.47: ResponseRequestQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
ReturnCode ?	integer	See Response/@ReturnCode .
Header	element	See Message/Header .
Notification ?	element	See Response/Notification .

7.14 Resource

The [Resource](#) message can be a [Command](#) message or a [Query](#) message to modify or to query **XJDF** resources. [QueryResource](#) retrieves information about the resources without modifying them, whereas [CommandResource](#) modifies those settings within the resource that is specified.

7.14.1 QueryResource

The [QueryResource](#) message retrieves information about resources and can be made selective by specifying a [ResourceQuParams](#) element. [QueryResource](#) can be used to retrieve information about either job-specific or global devices resources.

Table 7.48: QueryResource Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
Subscription ?	element	See Query/Subscription .
ResourceQuParams	element	Specifies the resources queried.

7.14.1.1 ResourceQuParams

Table 7.49: ResourceQuParams Element

NAME	DATA TYPE	DESCRIPTION
<i>ExternalID</i> ?	NMTOKEN	Resource / <i>@ExternalID</i> of the resource that is queried.
<i>JobID</i> ?	NMTOKEN	XJDF / <i>@JobID</i> for which resource information is being queried. If no <i>@JobID</i> is specified, the request applies to the currently running process or global resources, depending on the value of <i>@Scope</i> . <i>@JobID</i> SHALL NOT be specified if QueryResource/Subscription is present.
<i>JobPartID</i> ?	NMTOKEN	XJDF / <i>@JobPartID</i> 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.
<i>QueueEntryID</i> ?	NMTOKEN	QueueEntry / <i>@QueueEntryID</i> of the process that is currently being executed for which resource information is being queried. If <i>@QueueEntryID</i> is specified, <i>@JobID</i> , <i>@JobPartID</i> and Part SHALL NOT be specified. If none of <i>@JobID</i> , <i>@JobPartID</i> , Part or <i>@QueueEntryID</i> are specified, ResourceQuParams applies to all jobs. <i>@QueueEntryID</i> SHALL NOT be specified if QueryResource/Subscription is present.
<i>ResourceDetails</i> ?	enumeration	<i>@ResourceDetails</i> refines the level of information provided about the resources. Allowed values are: Brief – ResourceInfo/ResourceSet SHALL NOT contain the explicit resource elements as requested by <i>@ResourceName</i> . Full – ResourceInfo/ResourceSet SHALL contain the explicit resource elements as requested by <i>@ResourceName</i> .
<i>ResourceName</i> ?	NMTOKEN	If specified, ResourceInfo/ResourceSet / <i>@Name</i> SHALL match <i>@ResourceName</i> . Values include those from: ▶ Section 6 Resources.
<i>Scope</i>	enumeration	Specifies whether the Response or Signal SHALL return a complete list of all known resources, or the currently loaded resources or the resources related to a specific job. Allowed value is from: ▶ <i>Scope</i> .
Part *	element	Part elements that describe the resource whose messages are queried.

Example 7.4: Resource Query about Paper

The following is an example of an MIS sending a **QueryResource** to another MIS to get information on all paper known by the press.

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:19+00:00"/>
  <QueryResource>
    <Header DeviceID="TestSender" ID="Q1" Time="2018-02-28T16:00:19+00:00"/>
    <ResourceQuParams ResourceDetails="Full" ResourceName="Media" Scope="Allowed"/>
  </QueryResource>
</XJMF>
```

Example 7.5: Resource Response about Paper

The following is an example of a [ResponseResource](#) sent in response to the previous [QueryResource](#).

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="DeviceID" ID="1_000002" Time="2018-02-28T16:00:19+00:00"/>
  <ResponseResource>
    <Header DeviceID="DeviceID" ID="R1" Time="2018-02-28T16:00:19+00:00" refID="Q1"/>
    <ResourceInfo Scope="Allowed">
      <ResourceSet Name="Media">
        <Resource DescriptiveName="Paper # 1" ExternalID="ID_1">
          <Media Dimension="595.27559055 822.04724409" MediaType="Paper" Weight="80"/>
        </Resource>
        <Resource DescriptiveName="Paper # 2" ExternalID="ID_2">
          <Media Dimension="595.27559055 822.04724409" MediaType="Paper" Weight="100"/>
        </Resource>
        <!-- One Resource element for each paper follows here -->
      </ResourceSet>
    </ResourceInfo>
  </ResponseResource>
</XJMF>
```

7.14.2 CommandResource

The [CommandResource](#) message is used to modify or create global device databases such as media catalogs or lists of known machine operators.

Table 7.50: CommandResource Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
ResourceCmdParams	element	Specifies the resources to be modified.

7.14.2.1 ResourceCmdParams

Table 7.51: ResourceCmdParams Element

NAME	DATA TYPE	DESCRIPTION
UpdateMethod	enumeration	<p>@UpdateMethod specifies how the ResourceSet SHALL be updated.</p> <p>Allowed values are:</p> <p>Complete – Any resource selected by a Part SHALL be completely overwritten with the matching Resource from the ResourceSet in this message. Any Resource not selected by a Part SHALL not be modified.</p> <p>CompleteSet – The entire ResourceSet selected by this command SHALL be replaced by the ResourceSet this message contains.</p> <p>Incremental – Any Resource selected by a Part SHALL be incrementally updated with values from the matching Resource of the ResourceSet in this message whereby all items SHALL be added to the original Resource, replacing any previously existing matching items. Individual items not matched SHALL not be modified or removed.</p> <p>Remove – Any Resource selected by Part shall be completely removed from the ResourceSet. All other Resources SHALL NOT be modified or removed.</p> <p>RemoveSet – The entire ResourceSet selected by this message SHALL be removed.</p>
ResourceSet	element	ResourceSet defines the resources that are modified on the device according to the policy specified in @UpdateMethod .

Example 7.6: Resource Command: Uploading a list of paper Media

The following is an example of an MIS uploading a paper catalog to a device.

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:17+00:00"/>
  <CommandResource>
    <Header DeviceID="TestSender" ID="C1" Time="2018-02-28T16:00:17+00:00"/>
    <ResourceCmdParams UpdateMethod="Incremental">
      <ResourceSet Name="Media">
        <Resource DescriptiveName="Paper # 1" ExternalID="ID_1">
          <Media Dimension="595.27559055 822.04724409" MediaType="Paper" Weight="80"/>
        </Resource>
        <Resource DescriptiveName="Paper # 2" ExternalID="ID_2">
          <Media Dimension="595.27559055 822.04724409" MediaType="Paper" Weight="100"/>
        </Resource>
        <!-- One Resource element for each paper to upload follows here -->
      </ResourceSet>
    </ResourceCmdParams>
  </CommandResource>
</XJMF>
```

7.14.3 ResponseResource

When responding to [QueryResource](#), [ResponseResource](#) returns a [ResourceInfo](#) that contains the queried information concerning the resources.

Table 7.52: ResponseResource Message

NAME	DATA TYPE	DESCRIPTION
ReturnCode ?	integer	See Response/@ReturnCode . If ResourceInfo/ResourceSet is empty because the selective query parameters of the ResourceQuParams lead to a null selection of the known device or job resources, then @ReturnCode SHALL be one of 103 (@JobID unknown), 104 (@JobPartID unknown) or 108 (empty list) and SHOULD be flagged as an error with Notification[@Class="Error"] . When responding to an unsuccessful CommandResource , the value of @ReturnCode SHALL be set to "140" (CommandResource rejected). If the data could only be partially updated ResponseResource/@ReturnCode SHALL be "141". See ▶ Appendix B Return Values.
Header	element	See Message/Header .
Notification ?	element	See Response/Notification .
ResourceInfo *	element	Response to a CommandResource : Exactly one ResourceInfo SHALL be specified and SHALL contain information about the ResourceSet after modification. Response to a QueryResource : One ResourceInfo SHALL be specified for each ResourceSet that matches the QueryResource and SHALL contain information about the ResourceSet .

7.14.3.1 ResourceInfo

[ResourceInfo](#) SHALL specify the current state of a [ResourceSet](#) after any applicable modifications have been applied.

Table 7.53: ResourceInfo Element

NAME	DATA TYPE	DESCRIPTION
<i>CommandResult</i> ?	enumeration	Result of a <i>CommandResource</i> . Allowed values are: Merged – Values from the <i>ResourceSet</i> in <i>ResourceCmdParams</i> were merged into an existing <i>ResourceSet</i> . See the <i>ResourceInfo/ResourceSet</i> for the merged result. New – A new <i>ResourceSet</i> with the values specified in <i>ResourceCmdParams</i> was created. Rejected – The <i>CommandResource</i> was not applied to this <i>ResourceSet</i> . Removed – An existing <i>ResourceSet</i> was removed completely by a <i>ResourceSet</i> specified in <i>ResourceCmdParams</i> . Replaced – An existing <i>ResourceSet</i> was replaced completely by a <i>ResourceSet</i> specified in <i>ResourceCmdParams</i> .
<i>JobID</i> ?	NMTOKEN	@ <i>JobID</i> specifies 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>Level</i> ?	enumeration	Level of the consumable or output bin that is represented by this <i>ResourceInfo</i> for the device. If specified, exactly one <i>ResourceSet/Resource</i> SHALL be present. Allowed values are: Empty – The bin is empty. Full – The bin is full. High – The output bin is filling up and can soon be Full . This value is for output levels only and SHOULD NOT be specified for input resources. Low – The resources are running low and can soon be Empty . This value is for input levels only and SHOULD NOT be specified for output bins. OK – Specification is left to the device manufacturer.
<i>QueueEntryID</i> ?	NMTOKEN	@ <i>QueueEntryID</i> specifies the @ <i>QueueEntryID</i> of the queue entry that this <i>ResourceInfo</i> applies to.
<i>Scope</i> ?	enumeration	@ <i>Scope</i> specifies the context of the resources defined in this <i>ResourceInfo</i> . Allowed value is from: ▶ <i>Scope</i> .
<i>Speed</i> ?	float	The current speed at which the resource that this <i>ResourceInfo</i> describes is being consumed or produced. @ <i>Speed</i> SHALL be defined in the units specified by <i>ResourceSet/@Unit</i> / hour. If specified, exactly one <i>ResourceSet/Resource</i> SHALL be present.
<i>TotalAmount</i> ?	float	@ <i>TotalAmount</i> specifies the job independent total counter setting for a given type of resource. Note: This allows tracking of power consumption without requiring a device to track it individually for each job.
<i>MISDetails</i> ?	element	Definition of how the costs for the production of the <i>Resource</i> are to be charged.
<i>ResourceSet</i>	element	Additional information about the resource. If the <i>QueryResource</i> or <i>CommandResource</i> leading to this <i>ResourceInfo</i> contains <i>Part</i> elements, the <i>ResourceSet</i> SHALL contain no more than the appropriate matching <i>Resource</i> elements. Unless @ <i>Speed</i> or @ <i>Level</i> are present, <i>ResourceInfo</i> SHOULD contain all matching <i>Resource</i> elements.

7.14.4 SignalResource

SignalResource returns a **ResourceInfo** that contains the information concerning the subscribed resources.

Table 7.54: SignalResource Message

NAME	DATA TYPE	DESCRIPTION
<i>ChannelMode</i> ?	enumeration	Allowed value is from: ▶ ChannelMode. Note: See Signal/@ChannelMode .
<i>ReplaceAfter</i> ?	DateTime	The data from previous SignalResource messages in the same scope as specified by ResourceInfo/@Scope with the same Header/@DeviceID and a Header/@Time value after the time @ReplaceAfter and prior to the time specified by @ReplaceBefore (if present), SHALL be replaced by data in this SignalResource . If neither @ReplaceAfter or @ReplaceBefore is specified, this SignalResource is the original and SHALL NOT replace a previous SignalResource .
<i>ReplaceBefore</i> ?	DateTime	The data from previous SignalResource messages in the same scope as specified by ResourceInfo/@Scope with the same Header/@DeviceID and a Header/@Time value prior to the time @ReplaceBefore and after the time specified by @ReplaceAfter (if present), SHALL be replaced by data in this SignalResource . If neither @ReplaceAfter or @ReplaceBefore is specified, this SignalResource is the original and SHALL NOT replace a previous SignalResource .
Header	element	See Message/Header .
ResourceInfo *	element	ResourceInfo SHALL contain information concerning the subscribed resources.

Example 7.7: Resource Signal about Consumed Resources

The following is an example of a **Resource** signal used to report the consumption of paper **Media**.

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="DeviceID" ID="1_000002" Time="2018-02-28T16:00:20+00:00"/>
  <SignalResource>
    <Header DeviceID="DeviceID" ID="S1" Time="2018-02-28T16:00:20+00:00" refID="Sub1"/>
    <ResourceInfo JobID="Job1" JobPartID="Printing" Scope="Job">
      <ResourceSet Name="Media" Usage="Input">
        <Resource ExternalID="MIS-ID">
          <AmountPool>
            <PartAmount Amount="4500" Waste="66">
              <Part LotID="Lot1"/>
            </PartAmount>
            <PartAmount Amount="2200" Waste="22">
              <Part LotID="Lot2"/>
            </PartAmount>
          </AmountPool>
          <Part SheetName="S1"/>
        </Resource>
      </ResourceSet>
    </ResourceInfo>
  </SignalResource>
</XJMF>
```

7.15 ResubmitQueueEntry

A **QueueEntry** is resubmitted to a queue using the **ResubmitQueueEntry** message. This allows late changes to be made to an **XJDF** without affecting **QueueEntry** elements and their positions in a **Queue**. Resubmission modifies the **XJDF** with 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.

Note: See the **ModifyQueueEntry** command to modify the **QueueEntry** without modifying the underlying **XJDF** job.

7.15.1 CommandResubmitQueueEntry

Table 7.55: CommandResubmitQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See Message/Header .
<i>ResubmissionParams</i>	element	Defines the job resubmission.

7.15.1.1 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.

Table 7.56: ResubmissionParams Element

NAME	DATA TYPE	DESCRIPTION
<i>QueueEntryID</i>	NMTOKEN	QueueEntry/@QueueEntryID of the QueueEntry that is resubmitted SHALL match QueueEntry/@QueueEntryID of the queue entry to be replaced.
<i>UpdateMethod</i>	enumeration	<p>@UpdateMethod specifies how the QueueEntry SHALL be updated.</p> <p>Allowed values are:</p> <p>Complete – The QueueEntry SHALL be completely replaced by the XJDF that is referenced by @URL.</p> <p>Incremental – The QueueEntry SHALL be incrementally updated by the values of the XJDF that is referenced by @URL. All traits of the referenced XJDF are optional and only values that are explicitly specified in the referenced XJDF SHALL be modified. If XJDF/@JobPartID of the referenced XJDF is identical to XJDF/@JobPartID of the originally submitted XJDF or an XJDF that has been resubmitted with ResubmissionParams/@UpdateMethod="Incremental", then the process step that is identified by XJDF/@JobID and XJDF/@JobPartID SHALL be modified. Otherwise a new process step SHALL be submitted to the device.</p> <p>Remove – The QueueEntry SHALL be incrementally updated by removing all elements that are specified in the XJDF that is referenced by @URL. XJDF/@JobPartID of the referenced XJDF SHALL be identical to XJDF/@JobPartID of the originally submitted XJDF or an XJDF that has been resubmitted with ResubmissionParams/@UpdateMethod="Incremental".</p>
<i>URL</i>	URL	<p>Location of the XJDF to be submitted. XJDF/@JobID SHALL be identical to XJDF/@JobID of the originally submitted XJDF.</p> <p>If @URL refers to a directory, then all files with an extension of .xjdf that reside directly in the directory SHALL be processed in lexical order. The first XJDF is referred to as the primary XJDF. See ▶ Chapter 9 Referencing Multiple XJDF in a Directory.</p> <p>Note: A referenced directory MAY be inside a zip package. See ▶ [ZIP] for details.</p>

7.15.2 ResponseResubmitQueueEntry

Table 7.57: ResponseResubmitQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode .
<i>Header</i>	element	See Message/Header .
<i>Notification</i> ?	element	See Response/Notification .

7.16 ReturnQueueEntry

The **ReturnQueueEntry** message returns an **XJDF** that had been submitted with a **SubmitQueueEntry** to the controller that originally submitted the **XJDF**.

Note: 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**.

7.16.1 CommandReturnQueueEntry

Table 7.58: CommandReturnQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
ReturnQueueEntryParams	element	Defines the job being returned from device to controller after processing is completed or aborted.

7.16.1.1 ReturnQueueEntryParams

The **@URL** attribute specifies the location where the **XJDF** file to be returned can be retrieved by the controller. The scheme of the **@URL** attribute (such as "file", "http" or local url) SHALL define the retrieval method to be used to retrieve the **XJDF**.

Table 7.59: 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.

7.16.2 ResponseReturnQueueEntry

Table 7.60: ResponseReturnQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
ReturnCode ?	integer	See Response/@ReturnCode .
Header	element	See Message/Header .
Notification ?	element	See Response/Notification .

7.17 ShutDown

The **ShutDown** command message shuts down a controller or device. A device SHALL use the **Status** message if it signals its own shutdown.

7.17.1 CommandShutDown

Table 7.61: CommandShutDown Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
ShutDownCmdParams ?	element	Defines the details of a shutdown.

7.17.1.1 ShutDownCmdParams

Table 7.62: ShutDownCmdParams Element

NAME	DATA TYPE	DESCRIPTION
ShutDownType ?	enumeration	<p>Defines the device shutdown method.</p> <p>Allowed values are:</p> <p>Full – Completely shut down the device. It is no longer accessible via XJMF after the shutdown.</p> <p>StandBy – The device is set to standby mode. It can be restarted using a CommandWakeUp XJMF message. DevicelInfo/@StatusDetails SHOULD be "StandBy". If the device requires a CommandWakeUp XJMF prior to accepting new jobs, DevicelInfo/@Status SHALL be "Offline", else it SHALL be "Idle".</p>

7.17.2 ResponseShutDown

Table 7.63: ResponseShutDown Message

NAME	DATA TYPE	DESCRIPTION
ReturnCode ?	integer	See Response/@ReturnCode .
Header	element	See Message/Header .
Notification ?	element	See Response/Notification .

7.18 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 SHOULD contain a **DevicelInfo** element that contains the job-independent details of the device or controller and which MAY contain **JobPhase** elements that in turn contain the job specific information.

7.18.1 QueryStatus

Table 7.64: QueryStatus Message

NAME	DATA TYPE	DESCRIPTION
Header	element	See Message/Header .
Subscription ?	element	See Query/Subscription .
StatusQuParams ?	element	Acts as a filter to select the context of the device or controller that should be reported in the response messages' DevicelInfo element.

7.18.1.1 StatusQuParams

StatusQuParams is a filter that defines the job context for which information SHALL be returned in the response.

Table 7.65: StatusQuParams Element

NAME	DATA TYPE	DESCRIPTION
<i>QueueEntryID</i> ?	NMTOKEN	@ <i>QueueEntryID</i> of the queue entry that is being queried. If @ <i>QueueEntryID</i> is not specified, information SHALL be provided for all jobs (that are currently active) and job independent status. @ <i>QueueEntryID</i> SHALL NOT be specified if <i>QueryStatus/Subscription</i> is present.

7.18.2 ResponseStatus

Table 7.66: ResponseStatus Message

NAME	DATA TYPE	DESCRIPTION
<i>ReturnCode</i> ?	integer	See <i>Response/@ReturnCode</i> .
<i>Header</i>	element	See <i>Message/Header</i> .
<i>Notification</i> ?	element	See <i>Response/Notification</i> .
<i>DeviceInfo</i> ?	element	<i>DeviceInfo</i> describes details of the actual device status.

7.18.2.1 DeviceInfo

The response message returns a *DeviceInfo* element for the queried device. *Header/@DeviceID* SHALL specify the device that *DeviceInfo* describes.

Table 7.67: DeviceInfo Element (Sheet 1 of 2)

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: ▶ Units.
<i>HourCounter</i> ?	duration	The total integrated time (life time) of device operation in hours.
<i>IdleStartTime</i> ?	dateTime	@ <i>IdleStartTime</i> SHALL specify the time when the device switched to either @ <i>Status</i> ="Idle", @ <i>Status</i> ="Offline" or @ <i>Status</i> ="NonProductive". @ <i>IdleStartTime</i> SHALL NOT be specified if @ <i>Status</i> ="Production" or @ <i>Status</i> ="Stopped".
<i>ModuleIDs</i> ?	NMTOKENS	List of individual modules that are in use independent of a job. @ <i>ModuleIDs</i> SHALL not be specified for modules that are specified in <i>JobPhase/@Module</i> .
<i>PowerOnTime</i> ?	dateTime	Date and time when the device was switched on.
<i>ProductionCounter</i> ?	float	The current machine production counter. This counter can be reset manually. Typically, it starts counting at power-on time. The reset of this counter MAY be signaled by a <i>SignalNotification</i> with a <i>Notification/Event/@EventValue</i> ="CounterReset". The value of <i>Event/@EventID</i> for a counter reset is implementation specific. See ▶ Section 8.23.1 Event.
<i>Speed</i> ?	float	@ <i>Speed</i> specifies the current machine speed. @ <i>Speed</i> SHALL be defined in the same units as @ <i>CounterUnit</i> per hour.
<i>Status</i>	enumeration	@ <i>Status</i> describes the overall status of the device. Allowed value is from: ▶ DeviceStatus.
<i>StatusDetails</i> ?	NMTOKEN	String that defines the device state more specifically. Values include those from: ▶ Status Details.

Table 7.67: DeviceInfo Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
TotalProductionCounter?	float	The current total machine production counter since the machine was produced.
Activity*	element	Device and operator activities that are related to the device and are unrelated to a specific job.
FileSpec (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.
FileSpec (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.
JobPhase*	element	Each JobPhase SHALL describe the actual status of a job in the device. All jobs that are active on the device SHALL be specified. No JobPhase elements SHALL be present if @Status="Idle" or @Status="Offline". Multiple JobPhase elements specify that multiple job phases are active simultaneously on the device.

7.18.2.2 Activity

Activity elements allow tracking of device and operator tasks.

Table 7.68: 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.

7.18.3 JobPhase

JobPhase 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 an **AuditStatus** is that a **JobPhase** reflects a snapshot of the current job status whereas **AuditStatus** 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 7.69: JobPhase Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
Amount?	float	Total actual amount of good production that the process defined in this JobPhase produced since @StartTime. If @Waste is also specified, the value SHALL be without waste. The unit MAY be specified in the @CounterUnit attribute of the parent element DeviceInfo .
CostCenterID?	NMTOKEN	The cost center that this JobPhase job is currently being charged to.
DeadLine?	enumeration	Scheduling state of the job. Allowed values are: InTime – The job or job part will probably not miss the deadline. Late – The job or job part will miss the deadline. Warning – The job or job part could miss the deadline. Note: For more details on scheduling, see NodeInfo .

Table 7.69: JobPhase Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>JobID</i>	NMTOKEN	XJDF / <i>@JobID</i> of the process that is executing.
<i>JobPartID</i> ?	NMTOKEN	XJDF / <i>@JobPartID</i> of the process that is executing.
<i>ModuleIDs</i> ?	NMTOKENS	List of individual modules that are used to execute this JobPhase . <i>@ModuleIDs</i> SHALL NOT be specified for modules that are specified in <i>DevicelInfo</i> / <i>@ModuleIDs</i> .
<i>PercentCompleted</i> ?	float	Node processing progress in percent (%) completed.
<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.
<i>StartTime</i> ?	dateTime	Time when execution of the node that is described by this JobPhase has been started, defined by the transition of <i>NodeInfo</i> / <i>@Status</i> from "Waiting" to any active value.
<i>Status</i>	enumeration	<i>@Status</i> SHALL specify the <i>NodeInfo</i> / <i>@Status</i> of the process during this JobPhase . Allowed value is from: ▶ Status.
<i>StatusDetails</i> ?	NMTOKEN	Machine readable description that defines the job state more specifically. Values include those from: ▶ Status Details.
<i>Waste</i> ?	float	Total actual amount of waste that the process defined in this JobPhase produced since <i>@StartTime</i> . The unit MAY be specified in the <i>@CounterUnit</i> attribute of the parent element <i>DevicelInfo</i> .
<i>Activity</i> *	element	Device and operator activities that are related to a specific job or job phase.
<i>GangSource</i> *	element	If present, each GangSource SHALL represent the source jobs that are being processed as a gang job by this QueueEntry .
<i>MISDetails</i> ?	element	Definition of how the costs for this JobPhase are to be charged.
<i>Part</i> *	element	Part SHALL define which parts are currently being processed. The values should be a subset of <i>ResourceSet</i> [<i>@Name</i> ="NodeInfo"]/ <i>Resource</i> / <i>Part</i> of the XJDF that is processed during this JobPhase . For details on partitions, see ▶ Section 9.3.3 Partial Processing of XJDF with Partitioned ResourceSets.

7.18.4 SignalStatus

Table 7.70: SignalStatus Message (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ChannelMode</i> ?	enumeration	Allowed value is from: ▶ ChannelMode. Note: See <i>Signal</i> / <i>@ChannelMode</i> .
<i>ReplaceAfter</i> ?	DateTime	The data from previous SignalStatus messages with the same <i>Header</i> / <i>@DeviceID</i> and a <i>Header</i> / <i>@Time</i> after the time specified by <i>@ReplaceAfter</i> and prior to the time specified by <i>@ReplaceBefore</i> SHALL be replaced by data in this SignalStatus . <i>@ReplaceAfter</i> SHALL be specified if <i>@ReplaceBefore</i> is specified. If <i>@ReplaceAfter</i> and <i>@ReplaceBefore</i> are not specified, this SignalStatus is the original and SHALL NOT replace a previous SignalStatus .
<i>ReplaceBefore</i> ?	DateTime	The data from previous SignalStatus messages with the same <i>Header</i> / <i>@DeviceID</i> and a <i>Header</i> / <i>@Time</i> after the time specified by <i>@ReplaceAfter</i> and prior to the time specified by <i>@ReplaceBefore</i> SHALL be replaced by data in this SignalStatus . <i>@ReplaceBefore</i> SHALL be specified if <i>@ReplaceAfter</i> is specified. If <i>@ReplaceBefore</i> and <i>@ReplaceAfter</i> are not specified, this SignalStatus is the original and SHALL NOT replace a previous SignalStatus .

Table 7.70: SignalStatus Message (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See <i>Message/Header</i> .
<i>DeviceInfo</i>	element	<i>DeviceInfo</i> describes details of the actual device status.

Example 7.8: Status Signal

Example of two **XJMF** messages with *SignalStatus* elements. The first **XJMF** contains a heartbeat *SignalStatus* that was sent at 16:59 while the device was being setup. The second **XJMF** contains a *SignalStatus* that was sent at 17:00 as a result of a phase change when *JobPhase/@Status* went from "Setup" to "InProgress".

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="DeviceID" ID="1_000002" Time="2018-02-28T16:59:00+00:00"/>
  <SignalStatus>
    <Header DeviceID="DeviceID" ID="S1" Time="2018-02-28T16:59:00+00:00" refID="Sub1"/>
    <DeviceInfo Status="Production">
      <JobPhase JobID="j1" JobPartID="p1"
        StartTime="2018-02-28T16:00:00+00:00" Status="Setup"/>
    </DeviceInfo>
  </SignalStatus>
</XJMF>
```

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="DeviceID" ID="1_000004" Time="2018-02-28T17:00:00+00:00"/>
  <SignalStatus>
    <Header DeviceID="DeviceID" ID="S2" Time="2018-02-28T17:00:00+00:00" refID="Sub1"/>
    <DeviceInfo Status="Production">
      <JobPhase JobID="j1" JobPartID="p1"
        StartTime="2018-02-28T17:00:00+00:00" Status="InProgress"/>
    </DeviceInfo>
  </SignalStatus>
</XJMF>
```

7.19 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 to be unsubscribed by specifying a *StopPersChParams* element.

7.19.1 CommandStopPersistentChannel

Table 7.71: CommandStopPersistentChannel Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See <i>Message/Header</i> .
<i>StopPersChParams</i>	element	Specifies the persistent channel and the message types to be unsubscribed.

7.19.1.1 StopPersChParams

StopPersChParams provides a filter which selects persistent channels that SHALL be unregistered.

Table 7.72: StopPersChParams Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>ChannelID</i> ?	NMTOKEN	<i>Header/@refID</i> of the persistent channel to be deleted. <i>@ChannelID</i> specifies the <i>Header/@ID</i> of the <i>Query</i> message (identical to the <i>Header/@refID</i> of the <i>Signal</i> message).
<i>MessageType</i> ?	NMTOKEN	<i>@MessageType</i> SHALL match the local element name (i.e. without namespace prefix) of the <i>Signal</i> elements that SHALL be unregistered.

Table 7.72: StopPersChParams Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>URL</i> ?	URL	URL of the receiving controller. This SHALL be identical to the Subscription/@URL that was used to create the persistent channel.

7.19.2 ResponseStopPersistentChannel

Table 7.73: ResponseStopPersistentChannel Message

NAME	DATA TYPE	DESCRIPTION
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode .
<i>Header</i>	element	See Message/Header .
<i>Notification</i> ?	element	See Response/Notification .
<i>SubscriptionInfo</i> *	element	One SubscriptionInfo element SHALL be returned for every persistent channel that was removed.

7.20 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 associated with the submission.

[ResponseSubmitQueueEntry/QueueEntry/@QueueEntryID](#) SHALL be unique within the device. A new [@QueueEntryID](#) SHALL be generated for each [SubmitQueueEntry](#).

7.20.1 CommandSubmitQueueEntry

Table 7.74: CommandSubmitQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See Message/Header .
<i>QueueSubmissionParams</i>	element	Defines the job submission.

7.20.1.1 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** SHALL 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 7.75: QueueSubmissionParams Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Activation</i> ?	enumeration	Activation of the submitted QueueEntry . Allowed value is from: ▶ Activation.
<i>GangName</i> ?	NMTOKEN	Name of the gang for the job. If @GangName is specified, the QueueEntry SHOULD be executed along 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.
<i>GangPolicy</i> ?	enumeration	Ganging policy for the QueueEntry . Allowed value is from: ▶ GangPolicy.
<i>NextQueueEntryID</i> ?	NMTOKEN	QueueEntry/@QueueEntryID of the queue entry that SHALL be positioned directly behind the entry. At most one of @NextQueueEntryID , @PrevQueueEntryID or @Priority SHALL be specified.

Table 7.75: QueueSubmissionParams Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>PrevQueueEntryID</i> ?	NMTOKEN	QueueEntry/@QueueEntryID of the queue entry that SHALL be positioned directly in front of the entry. At most one of @NextQueueEntryID , @PrevQueueEntryID or @Priority SHALL be specified.
<i>Priority</i> ?	integer	Number from 0 to 100, where "0" is the lowest priority and "100" is the maximum priority. At most one of @NextQueueEntryID , @PrevQueueEntryID or @Priority SHALL be specified. Note that QueueSubmissionParams/@Priority is not the same as NodeInfo/@JobPriority . QueueSubmissionParams/@Priority specifies the priority in the context of the device queue whereas NodeInfo/@JobPriority specifies the priority of the task in general. QueueSubmissionParams/@Priority MAY be modified due to additional scheduling information (e.g., NodeInfo/@FirstStart). QueueSubmissionParams/@Priority and ModifyQueueEntryParams/@Priority SHALL take precedence over NodeInfo/@JobPriority .
<i>ReturnJMF</i> ?	URL	Address queue where a ReturnQueueEntry Command SHALL be sent when the QueueEntry is completed or aborted.
<i>URL</i>	URL	Location of the XJDF to be submitted or resubmitted. If @URL refers to a directory, then all files with an extension of 'xjdf' that reside directly in the directory SHALL be processed in lexical order. The first XJDF is referred to as the primary XJDF . See ▶ Section 9.4.1 Referencing Multiple XJDF in a Directory. Note: A referenced directory MAY be inside a zip package. Refer to the application note ▶ [ZIP].

Example 7.9: SubmitQueueEntry Command 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.

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:21+00:00"/>
  <CommandSubmitQueueEntry>
    <Header DeviceID="TestSender" ID="C1" Time="2018-02-28T16:00:21+00:00"/>
    <QueueSubmissionParams URL="http://jobserver.xjdf.org?job1"/>
  </CommandSubmitQueueEntry>
</XJMF>
```

7.20.2 ResponseSubmitQueueEntry

Table 7.76: ResponseSubmitQueueEntry Message

NAME	DATA TYPE	DESCRIPTION
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode .
<i>Header</i>	element	See Message/Header .
<i>Notification</i> ?	element	See Response/Notification .
<i>QueueEntry</i> ?	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.

7.21 WakeUp

The **WakeUp** command message activates a controller or device that has been in stand-by mode. All [QueueEntry](#) elements SHALL have [@Activation="Held"](#) and SHALL be explicitly resumed with a [ModifyQueueEntry](#) with a **Resume** operation. A device SHALL use the **Status** message to signal its own awakening.

7.21.1 CommandWakeUp

Table 7.77: CommandWakeUp Message

NAME	DATA TYPE	DESCRIPTION
<i>Header</i>	element	See Message/Header .

7.21.2 ResponseWakeUp

Table 7.78: ResponseWakeUp Message

NAME	DATA TYPE	DESCRIPTION
<i>ReturnCode</i> ?	integer	See Response/@ReturnCode .
<i>Header</i>	element	See Message/Header .
<i>Notification</i> ?	element	See Response/Notification .

8 Subelements

The elements in this chapter are subelements that can occur in multiple elements. They are not specific resource elements and are therefore never directly linked to processes.

8.1 Address

Definition of an address. The structure is derived from the vCard format. The corresponding vCard fields are quoted in the table.

Element Properties

Element referenced by: [Contact](#)

Table 8.1: Address Element

NAME	DATA TYPE	DESCRIPTION
<i>AddressUsage</i> ?	NMTOKEN	@AddressUsage specifies the intended use of the address. Values include: Business - Business company address. Residence - Private home address.
<i>City</i> ?	string	City or locality of the Address (vCard: ADR:locality).
<i>CivicNumber</i> ?	string	@CivicNumber SHALL specify the street number of the street address. If @CivicNumber is specified, it SHALL NOT be included in @Street.
<i>Country</i> ?	string	Country code of the Address (vCard: ADR:country).
<i>CountryCode</i> ?	NMTOKEN	Country of the Address . Values include those from: ▶ [ISO/NP 3166-1:2013].
<i>ExtendedAddress</i> ?	string	Extended address (vCard: ADR:extadd. For example: Suite 245).
<i>PostalCode</i> ?	string	Zip code or postal code of the 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 of the Address (vCard: ADR:region).
<i>Street</i> ?	string	Street of the Address (vCard: ADR:street). @Street SHALL include the name of the street and SHOULD include the street number unless the street number is specified separately in @CivicNumber.
AddressLine *	element	Each AddressLine element SHALL specify one line of a printed address. If AddressLine is provided, the complete address SHALL be provided as an ordered sequence of AddressLine elements.

8.1.1 AddressLine

AddressLine represents an individual address line.

Note: An address may be encoded as attributes (e.g. @City, @Street, etc), text elements in **AddressLine**, or both. The latter case may occur where the original database does not provide all the individual details of the address.

Table 8.2: AddressLine Element

NAME	DATA TYPE	DESCRIPTION
	text	Text content of the AddressLine .

8.2 ApprovalPerson

ApprovalPerson specifies the details of the person who is responsible for signing an approval.

Element Properties

Element referenced by: [ApprovalDetails](#), [ApprovalParams](#)

Table 8.3: ApprovalPerson Element

NAME	DATA TYPE	DESCRIPTION
ApprovalRole ?	enumeration	Role of the ApprovalPerson . Allowed values are: Approvinator – 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. Informative – The approver is informed of the Approval process, but the approval is still valid, even without his approval. Obligated – The approver SHALL sign the approval.
ApprovalRoleDetails ?	string	Additional details on the @ApprovalRole . @ApprovalRole SHOULD be specified if @ApprovalRoleDetails is provided.
ContactRef	IDREF	Additional details of the person who SHALL sign the approval. The referenced Contact SHALL contain a Part / @ContactType="Approver" .

8.3 AutomatedOverPrintParams

AutomatedOverPrintParams provides controls for the automated selection of overprinting of black text or graphics.

Element Properties

Element referenced by: [RenderingParams](#), [SeparationControlParams](#)

Table 8.4: AutomatedOverPrintParams Element

NAME	DATA TYPE	DESCRIPTION
KnockOutCMYKWhite ?	boolean	If @KnockOutCMYKWhite="true" , 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.
LineArtBlackLevel ?	float	A value between 0.0 and 1.0 that indicates the minimum black level for the stroke or fill colors that cause the line art to be set to overprint. @LineArtBlackLevel SHALL NOT be specified unless @OverPrintBlackLineArt="true" .
OverPrintBlackLineArt ?	boolean	Indicates whether overprint SHALL 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.
OverPrintBlackText ?	boolean	Indicates whether overprint SHALL be set to "true" for black text. If "true", overprint of black text is applied regardless of any values in the PDL.
TextBlackLevel ?	float	A value between 0.0 and 1.0 that indicates the minimum black level for the text stroke or fill colors that cause the text to be set to overprint. @TextBlackLevel SHALL NOT be specified unless @OverPrintBlackText="true" .
TextSizeThreshold ?	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 SHALL be used. @TextSizeThreshold SHALL NOT be specified unless @OverPrintBlackText="true" .

8.4 BarcodeCompParams

BarcodeCompParams specifies the technical compensation parameters for barcodes.

Element Properties

Element referenced by: [BarcodeReproParams](#)

Table 8.5: BarcodeCompParams Element

NAME	DATA TYPE	DESCRIPTION
<i>CompensationProcesses</i>	enumeration	Process that is bar width spread SHALL be compensated for. Allowed values are: Platemaking Printing
<i>CompensationValue</i>	float	The width of the bars SHALL be reduced by this amount (in microns) to compensate for technical spread.

8.5 BarcodeReproParams

BarcodeReproParams specifies the reproduction parameters for barcodes.

Element Properties

Element referenced by: *Content/BarcodeProductionParams*, *Layout/StripMark*

Table 8.6: BarcodeReproParams Element

NAME	DATA TYPE	DESCRIPTION
<i>BearerBars</i> ?	enumeration	@ <i>BearerBars</i> specifies how to generate bearer bars. (ITF). Allowed values are: Box BoxHMarks None TopBottom
<i>Height</i> ?	float	@ <i>Height</i> SHALL specify the height (Y direction) of the bars of a linear barcode in the PDL.
<i>Magnification</i> ?	float	The magnification factor that linear barcodes SHALL be scaled with. For example, a value for @ <i>Magnification</i> > 1 requests thicker barcode lines in the resulting PDL.
<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, the barcode is put on top of underlying graphics. <i>WhiteBox</i> – An area of the underlying graphics SHALL be masked out (the white box) and the barcode SHALL be put on top of this masked area. The area of the white box SHALL be the box enclosing all artwork of the barcode, excluding optional human readable text. The box SHALL enclose bearer bars, quiet zones and non-optional human readable text (UPC and EAN barcodes).
<i>ModuleHeight</i> ?	float	The Y size in microns of an element of a 2D barcode (e.g., PDF417). For DATAMATRIX, Y dimension MAY be omitted (X dimension = Y dimension).
<i>ModuleWidth</i> ?	float	The X size in microns of an element of a 2D barcode such as DATAMATRIX or PDF417.
<i>Ratio</i> ?	float	The ratio between the width of the narrow bars and the wide bars for those barcodes where the ratio of the width of the wide bars to the narrow bars MAY vary.
<i>BarcodeCompParams</i> *	element	Parameters for bar width compensation. The total reduction of bar width SHALL be the sum of all <i>BarcodeCompParams</i> /@ <i>CompensationValue</i> .

8.6 Certification

Certification specifies the certification properties of paper.

Element Properties

Element referenced by: [MediaIntent](#), [Media](#)

Table 8.7: Certification Element

NAME	DATA TYPE	DESCRIPTION
Claim ?	string	Name of the certification as defined by the issuing organization. Values include: FSC 100% FSC Mix 70% FSC Mix Credit FSC Recycled 85% FSC Recycled Credit PEFC nn% PEFC Certified PEFC Recycled
Identifier ?	string	Certification identification number as defined by the issuing organization.
Organization ?	NMTOKEN	Identifier of the issuing organization: Values include: CFCC – China's National Forest Certification System FSC – Forest Stewardship Council IFCC – Sustainable Forest Management Requirements PEFC – The Programme for the Endorsement of Forest Certification

8.7 Comment

The [Comment](#) element can be used to provide human readable text.

Element Properties

Element referenced by: [XJDF](#), [ResourceSet](#), [Resource](#), [Notification](#), [ContentMetadata](#), [PreflightCheck](#)

Table 8.8: Comment Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
Author ?	string	Human readable text that identifies the person who created the Comment . See also @PersonalID .
ExternalID ?	NMTOKEN	Identification that is used to reference the Comment .
Language ?	language	Human readable language of the Comment .
PersonalID ?	NMTOKEN	Machine readable identifier of the employee that entered the comment. When the Comment is created by a person with a known Contact / @UserID , then @PersonalID SHOULD contain the value of Contact / @UserID . See also @Author .
TimeStamp ?	dateTime	Describes the date and time when the Comment was created.

Table 8.8: Comment Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Type</i> ?	NMTOKEN	<p>@<i>Type</i> specifies the usage of a comment.</p> <p>Values include:</p> <p>DeviceText – Human readable description created by the device that provides details beyond the value of @<i>StatusDetails</i>. The "DeviceText" value SHOULD only be specified in <i>Notification/Comment/@Type</i>.</p> <p>Instruction – Message to the operator that contains information regarding the processing of the job. The "Instruction" value SHOULD only be specified in <i>XJDF/Comment/@Type</i>.</p> <p>JobDescription – Description of the job. The "JobDescription" value SHOULD only be specified in <i>XJDF/Comment/@Type</i>. See also <i>CustomerInfo/@CustomerJobName</i>.</p> <p>OperatorText – Message from the operator that contains information regarding the processing of the job. The "OperatorText" value SHOULD only be specified in <i>Notification/Comment/@Type</i>.</p> <p>Orientation – Description of the orientation of a <i>Resource</i>. The "Orientation" value SHOULD only be specified in <i>Resource/Comment/@Type</i> or <i>ResourceSet/Comment/@Type</i>.</p>
	text	<p>Body of the comment.</p> <p>Note: Whitespace is preserved only as generic whitespace in XML. Applications that display comments to the user SHOULD maintain whitespace.</p>

Example 8.1: Multi-line Comment

The following example shows a multi-line comment with whitespace.

```
<Comment ExternalID="c_000004" Type="Instruction">Multiline text
  with white space
```

```
and empty lines</Comment>
```

8.8 ConvertingConfig

The *ConvertingConfig* element describes a range of sheet sizes that can be used for optimizing a die layout in *DieLayoutProduction* or a press sheet for *SheetOptimizing*.

Element Properties

Element referenced by: *DieLayoutProductionParams*, *SheetOptimizingParams*

Table 8.9: ConvertingConfig Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>MarginBottom</i> ?	float	The bottom margin for positioning the layout on the sheet.
<i>MarginLeft</i> ?	float	The left margin for positioning the layout on the sheet.
<i>MarginRight</i> ?	float	The right margin for positioning the layout on the sheet.
<i>MarginTop</i> ?	float	The top margin for positioning the layout on the sheet.
<i>SheetHeightMax</i> ?	float	The maximum sheet height, in points.
<i>SheetHeightMin</i> ?	float	The minimum sheet height, in points.
<i>SheetWidthMax</i> ?	float	The maximum sheet width, in points.
<i>SheetWidthMin</i> ?	float	The minimum sheet width, in points.
<i>Device</i> *	element	The target devices (printing press, die cutter and further finishing equipment) corresponding to this configuration. Typically only the type of <i>Device</i> would be used (e.g., the model of the die cutter). If multiple <i>Devices</i> are specified, then the other attributes in this element SHALL apply to a production configuration that uses all specified <i>Devices</i> .

Table 8.9: ConvertingConfig Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Media</i> *	element	Zero or more <i>Media</i> elements that are candidates for optimization. Note: <i>Media</i> allows a media database savvy consumer to loop over an explicit list of known materials rather than providing results based on a range of dimensions only.

8.9 Crease

Crease defines an individual crease line on a component.

Element Properties

Element referenced by: [CreasingParams](#), [FoldingParams](#)

Table 8.10: Crease Element

NAME	DATA TYPE	DESCRIPTION
<i>Depth</i> ?	float	Depth of the crease, measured in microns [μm].
<i>StartPosition</i> ?	XYPair	Starting position of the tool.
<i>WorkingDirection</i> ?	enumeration	Direction from which the tool is working. Allowed value is from: ▶ WorkingDirection .
<i>WorkingPath</i> ?	XYPair	Working path of the tool beginning at @StartPosition .

8.10 Cut

Cut describes one straight cut with an arbitrary tool.

Element Properties

Element referenced by: [CuttingParams](#), [FoldingParams](#)

Table 8.11: Cut Element

NAME	DATA TYPE	DESCRIPTION
<i>CutWidth</i> ?	float	Width in points of u-shaped knife, saw blade, etc.
<i>StartPosition</i> ?	XYPair	Starting position of the tool.
<i>WorkingDirection</i> ?	enumeration	Direction from which the tool is working. Allowed value is from: ▶ WorkingDirection .
<i>WorkingPath</i> ?	XYPair	Working path of the tool beginning at @StartPosition .

8.11 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.

Element Properties

Element referenced by: [ApprovalDetails](#), [ColorSpaceConversionOp](#), [ColorSpaceConversionParams](#), [Device/IconList/Icon](#), [DieLayout](#), [LayoutElementProductionParams](#), [Preview](#), [QualityControlResult](#), [RunList](#),

*ShapeDef, ShapeDefProductionParams/ObjectModel, ShapeDefProductionParams/
ShapeTemplate*

Table 8.12: FileSpec Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Checksum</i> ?	hexBinary	Checksum of the file being referenced using the RSA MD5 algorithm. 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>Encoding</i> ?	NMTOKEN	Encoding or code page of the file contents. Values include those from: ▶ [IANA-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. 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. Allowed values are from: ▶ Appendix E String Generation.
<i>FileSize</i> ?	integer	Size of the file in bytes.
<i>FileTemplate</i> ?	NMTOKENS	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. 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. Allowed values are from: ▶ Appendix E String Generation.
<i>MimeType</i> ?	string	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). 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.
<i>OverwritePolicy</i> ?	enumeration	Policy that specifies the policy to follow when a file already exists and the <i>FileSpec</i> is used in an output resource. Allowed values are: <i>Abort</i> – Abort the process without modifying the old file. <i>NewVersion</i> – Create a new file version. Only valid when the <i>FileSpec</i> references a file on a version aware file system. <i>OperatorIntervention</i> – Present a dialog to an operator. <i>Overwrite</i> – Overwrite the old file. <i>RenameNew</i> – Rename the new file. <i>RenameOld</i> – Rename the old file.
<i>Password</i> ?	string	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.
<i>ResourceUsage</i> ?	NMTOKEN	If this specification specifies <i>FileSpec(ResourceUsage)</i> in the name column of an element table, then <i>FileSpec/@ResourceUsage</i> SHALL be provided. See ▶ Table 1.3 Template for Element Descriptions for details. Note: <i>@ResourceUsage</i> is generally required if an element contains more than one <i>FileSpec</i> subelement.
<i>SearchDepth</i> ?	integer	Used when <i>FileSpec</i> 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.

Table 8.12: FileSpec Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>UID</i> ?	NMTOKEN	Internal ID of the referenced file. The <i>@UID</i> SHALL be unique within the workflow. The value of <i>@UID</i> is dependent on the type of file that is referenced: <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. If neither <i>@URL</i> nor <i>@UID</i> is present on an input <i>FileSpec</i> , 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.
<i>URL</i> ?	URL	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 <i>FileSpec</i> , 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. See ▶ [RFC3986] for the syntax and examples. For the ‘file’ URL scheme see also ▶ [RFC1738].
<i>UserFileName</i> ?	string	A user-friendly name that 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 <i>FileSpec</i> completes. If not specified the file specified by this <i>FileSpec</i> SHOULD NOT be deleted by the device.

8.11.1 Disposition

This element describes how long the digital asset that is referenced by the *FileSpec* SHOULD be maintained by a device. The device SHALL perform an action defined by *Disposition/@DispositionAction* when a “disposition time” occurs. Disposition time is defined as either:

- $@Until \leq \text{"Disposition time"} \leq @Until + @ExtraDuration$
- $ProcessCompleteTime + @MinDuration \leq \text{"Disposition time"} \leq ProcessCompleteTime + @MinDuration + @ExtraDuration$

Table 8.13: Disposition Element

NAME	DATA TYPE	DESCRIPTION
<i>DispositionAction</i> ?	enumeration	<i>@DispositionAction</i> specifies the required disposal action for the asset. Allowed values are: <i>Archive</i> – The asset SHALL be archived when disposition time occurs. <i>Delete</i> – The asset SHALL be deleted when disposition time occurs.
<i>ExtraDuration</i> ?	duration	Indicates the maximum duration that the device SHALL 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 MAY be retained for a system specified time.
<i>MinDuration</i> ?	duration	Indicates the minimum duration for which the device SHOULD retain the asset after the process that uses the asset completes. <i>@MinDuration</i> SHALL NOT be specified if <i>@Until</i> is present.
<i>Priority</i> ?	integer	Value between 0 and 100 that specifies the order in which assets SHALL be 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> SHALL be deleted first.
<i>Until</i> ?	dateTime	Indicates an absolute point in time when the device or application SHOULD discard the asset. <i>@Until</i> SHALL NOT be specified if <i>@MinDuration</i> is present.

8.12 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 containers are.

Element Properties

Element referenced by: [InterpretingParams](#), [Layout](#), [RasterReadingParams](#)

Table 8.14: FitPolicy Element

NAME	DATA TYPE	DESCRIPTION
ClipOffset ?	XYPair	Defines the offset (position) of the imaged area in the non-rotated source image when @SizePolicy 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 SHALL be taken from the center of the source image.
GutterPolicy ?	enumeration	Allows printing of NUp grids even if the media size does not match the requirements of the data. Allowed values are: Distribute – The gutters can grow or shrink to the value specified in @MinGutter . Fixed – The gutters are fixed.
MinGutter ?	XYPair	Minimum width in points of the horizontal and vertical gutters formed between rows and columns of pages of a multi-up sheet layout. The first value specifies the minimum width of all horizontal gutters and the second value specifies the minimum width of all vertical gutters.
RotatePolicy ?	enumeration	Specifies the policy for the device to automatically rotate the content to optimize the fit of the content to the receiving container. Allowed values are: NoRotate – Do not rotate. RotateClockwise – Rotate clockwise by 90°. RotateCounterClockwise – Rotate counterclockwise by 90°. RotateOrthogonal – Rotate by 90° in either direction.
SizePolicy ?	enumeration	Allows printing even if the container size does not match the requirements of the data. Allowed values are: Abort – Emit an error and abort printing. ClipToMaxPage – The page contents SHALL be clipped to the size of the container. The printed area is either centered in the source image if no @ClipOffset key is given, or from that position that is determined by @ClipOffset . FitToPage – The page contents SHALL be scaled up or down to fit the container. The aspect ratio SHALL be maintained. ReduceToFit – The page contents SHALL be scaled down but not scaled up to fit the container. The aspect ratio SHALL be maintained. Tile – The page contents SHALL be split into several tiles, each tile SHALL be printed on its own surface.

8.13 Fold

[Fold](#) describes an individual folding operation of the [Component](#).

Element Properties

Element referenced by: [FoldingIntent](#), [FoldingParams](#)

Table 8.15: Fold Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
From	enumeration	Edge from which the page SHALL be folded. Allowed values are: Front Left

Table 8.15: Fold Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>To</i>	enumeration	Direction in which the page SHALL be folded. Allowed values are: Up – Upwards; corresponds to a valley fold with the left/bottom side coming over the opposite side. Down – Downwards; corresponds to a mountain or peak fold with the left/bottom side coming under the opposite side.
<i>Travel ?</i>	float	Distance of the reference edge relative to <i>@From</i> .

8.14 GangSource

GangSource provides source job information about a **BinderySignature** that is placed on a gang form.

Element Properties

Element referenced by: **JobPhase, QueueFilter, QueueEntry, NodeInfo**

Table 8.16: GangSource Element

NAME	DATA TYPE	DESCRIPTION
<i>BinderySignatureID ?</i>	NMTOKEN	If present, <i>@BinderySignatureID</i> SHALL reference the BinderySignature that this GangSource represents.
<i>Copies</i>	integer	<i>@Copies</i> SHALL specify the number of copies of the BinderySignature that are required.
<i>JobID</i>	NMTOKEN	<i>@JobID</i> SHALL reference XJDF/@JobID of the individual job that describes the processing prior to and after printing and cutting the gang sheet.

8.15 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.

Element Properties

Element referenced by: **XJDF, ResourceSet, Product, Resource, Content/ContentMetadata, PreflightParams/PreflightTest, PreflightReport/PreflightCheck**

Table 8.17: GeneralID Element

NAME	DATA TYPE	DESCRIPTION
<i>DataType ?</i>	enumeration	Data type of the variable. Allowed value is from: ▶ <i>DataType</i> .
<i>IDUsage</i>	NMTOKEN	Usage of the GeneralID .
<i>IDValue</i>	string	Value of the GeneralID . The data type of the value SHALL correspond to GeneralID/@DataType .

8.16 Glue

This element provides the information for determining 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: [AssemblingIntent/BindIn](#), [AssemblingIntent/StickOn](#), [BoxFoldingParams](#), [CaseMakingParams](#), [EndSheetGluingParams](#), [GluingParams](#), [HeadBandApplicationParams](#), [InsertingParams](#), [ThreadSewingParams](#), [MediaLayers](#)

Table 8.18: Glue Element

NAME	DATA TYPE	DESCRIPTION
AreaGlue ?	boolean	Specifies that this Glue SHOULD cover the complete width of the Component it is applied to.
GlueLineWidth ?	float	Width of the glue line in points. If not specified, the default behavior depends on the value of @AreaGlue : If @AreaGlue = "true", then the implied width is the width of the Component . If @AreaGlue = "false", then the implied width is the system dependent glue line width.
GlueRef ?	IDREF	Reference to a MiscConsumable that represents the physical glue.
GlueType ?	enumeration	Glue type. Allowed values are: ColdGlue – Any type of glue that needs no heat treatment. Hotmelt – Hotmelt EVA (Ethylene-vinyl acetate). Permanent – Any glue that is designed not to be removed. PUR – Polyurethane. Removable – Any glue that is designed to be removed.
GluingPattern ?	FloatList	Glue line pattern defined by the length of a glue line segment (1st element, 3rd and all odd elements of the list of values) and glue line gap (2nd element, 4th and all even elements of the list of values). A solid line SHALL be expressed by the pattern (1 0). @GluingPattern SHALL contain an even number of entries. If the total length of @GluingPattern is less than @WorkingPath , the pattern restarts after the last gap. If the total length of @GluingPattern is larger than @WorkingPath , the pattern SHALL be clipped at the end.
GluingTechnique ?	enumeration	When glue is specified in the context of hardcover binding, then @GluingTechnique specifies the technique of gluing operation. Allowed values are: SideGluingBack SideGluingFront SpineGluing
MeltingTemperature ?	integer	Temperature needed for melting the glue, in degrees centigrade. @MeltingTemperature SHALL NOT be specified unless @GlueType ="Hotmelt" or @GlueType ="PUR".
StartPosition ?	XYPair	Start position of the glue line.
WorkingDirection ?	enumeration	Direction from which the glue should be applied to the Component . Allowed value is from: ▶ Face.
WorkingPath ?	XYPair	Relative working path of the gluing tool.

8.17 HolePattern

The [HolePattern](#) element describes a pattern of one or more holes.

Note: For dealing with the default case of [@HoleCount](#) (i.e., when it is not supplied), intelligent systems MAY take into consideration physical properties such as 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 SHALL match 100% between the [HoleMaking](#) and the process for obvious reasons.

Element Properties

Element referenced by: [HoleMakingIntent](#), [HoleMakingParams](#)

Table 8.19: HolePattern Element

NAME	DATA TYPE	DESCRIPTION
Center ?	XYPair	Position of the center of the first hole relative to the coordinate system that is defined in @CenterReference or the coordinate system of the input Component if @CenterReference is not present. If not specified, the value SHALL be defined by the value of @Pattern .
CenterReference ?	enumeration	Defines the reference coordinate system for @Center . Allowed values are: RegistrationMark – The center is relative to a registration mark. TrailingEdge – Physical coordinate system of the component. Note: RegistrationMark is typically used in webfed printing where no trailing edge is available.
Extent ?	XYPair	Size (bounding box) of each hole, in points. If @Shape is "Round", only the first entry of @Extent SHALL be evaluated and SHALL define the hole diameter. If not specified, the value SHALL be defined by the value of @Pattern .
HoleCount ?	IntegerList	@HoleCount 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: "2 2 2" = "h h s s h h". "0 3 3 3 3" = "s s s h h h s s s h h h". Note: @HoleCount is typically applied to patterns with @Pattern whose enumeration values begin with a "P", "W" or "C" in ▶ Table G.1 Naming Scheme for Hole Patterns.
Pattern ?	NMTOKEN	Predefined hole pattern. @Pattern SHALL be supplied if one of @Center , @Extent or @Shape is not specified. Allowed values is from: ▶ Section G Hole Pattern Catalog.
Pitch ?	XYPair	If @Pitch is specified, this HolePattern represents a line of holes. @Pitch represents the distance between the centers of two adjacent holes.
ReferenceEdge ?	enumeration	The edge of the Component relative to where the holes SHALL be placed. Allowed values are: Bottom Left Pattern – Specifies the reference edge implied by the value of @Pattern in ▶ Section G Hole Pattern Catalog. Right Top
Reinforcement ?	NMTOKEN	@Reinforcement specifies how the holes SHALL be reinforced. Values include: Grommet Note: Additional details of the reinforcement MAY be supplied in a MiscConsumable with MiscConsumable/@Type="Grommet" .
Shape ?	enumeration	Shape of the holes. If not specified, the value SHALL be defined by the value of @Pattern . Allowed values are: Elliptic Rectangular Round

8.18 IdentificationField

This resource contains information about a mark on a document, e.g. a bar code. The data in [IdentificationField](#) can be used to dynamically generate barcodes. It can also be used to decode the contents of a bar code, e.g. when used for OCR-based verification purposes or document separation.

Element Properties

Element referenced by: [Component](#), [Content/BarcodeProductionParams](#), [Device](#), [EmbossingParams/Emboss](#), [ExposedMedia](#), [Ink](#), [Layout/StripMark](#), [Media](#), [MiscConsumable](#), [Pallet](#), [Tool](#), [Module](#)

Table 8.20: IdentificationField Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
BoundingBox ?	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 SHALL be defined by the StripMark containing the IdentificationField. If no Layout context is available, the origin of the coordinate system SHALL be defined as the lower left corner of the resource surface that @Position specifies when the specified surface is viewed in its natural orientation.</p> <p>Each item in the list below specifies a value of @Position 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 @Position = "Left", the origin is the bottom-back corner of the left surface when viewed from the front surface of the resource and lower left corner when viewed from the left surface.</p> <p>"Back" – Bottom right corner. "Bottom" – Back left corner. "Front" – Bottom left corner. "Left" – Bottom back corner. "Right" – Bottom front corner. "Top" – Front left corner.</p> <p>If no @BoundingBox is defined and the IdentificationField is specified outside the context of a Layout, the complete visible surface SHALL be scanned for an appropriate bar code.</p> <p>If no @BoundingBox is defined and the IdentificationField is specified within the context of a Layout, the implied @BoundingBox SHALL be specified by the position of the StripMark.</p> <p>Note: @BoundingBox is used only as metadata when searching or scanning IdentificationField elements and not used when generating IdentificationField elements in a LayoutElementProduction process.</p>
Encoding ?	enumeration	<p>Encoding of the information.</p> <p>Allowed values are:</p> <p>ASCII – Plain-text font. Barcode – Any bar code. Braille – Braille text. RFID – Radio Frequency Identification tag.</p>
EncodingDetails ?	NMTOKEN	<p>Details about the encoding type. An example is the bar code scheme.</p> <p>Values include those from: ▶ Table 8.21 EncodingDetails Attribute Values.</p>
Format ?	regExp	<p>Regular expression that defines the expected format of the expression (e.g., the number of digits, alphanumeric or numeric).</p> <p>Note: This field MAY also be used to define constant fields (e.g., the end of document markers or packaging labels).</p> <p>If not specified, any expression is valid. Exactly one of @Format, @Value or the pair @ValueFormat and @ValueTemplate SHALL be specified.</p>
Orientation ?	matrix	<p>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 @Orientation is used only as metadata when searching or scanning IdentificationField elements and not used when generating IdentificationField elements in a LayoutElementProduction process.</p>
Position ?	enumeration	<p>Position with respect to the instance document or Resource to which IdentificationField refers.</p> <p>Allowed value is from: ▶ Face.</p>

Table 8.20: IdentificationField Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Purpose</i> ?	enumeration	Purpose defines the usage of the field. Allowed values are: Label – Used to mark a product or component. Separation – Used to separate documents. Verification – Used for verification of documents.
<i>PurposeDetails</i> ?	NMTOKEN	More detail about the usage of the barcode. Values include: ProductIdentification – End product identification (e.g., scanning in the super-market).
<i>Value</i> ?	string	Fixed value of the <i>IdentificationField</i> (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. Exactly one of <i>@Format</i> , <i>@Value</i> or the pair <i>@ValueFormat</i> and <i>@ValueTemplate</i> SHALL be specified. Allowed values are from: ▶ Appendix E String Generation.
<i>ValueTemplate</i> ?	NMTOKENS	A list of values used with <i>@ValueFormat</i> to define fixed and/or variable content of barcodes or text. If <i>MetadataMap</i> elements are present, <i>MetadataMap/@Name</i> SHALL be included in <i>@ValueTemplate</i> to select the data from the <i>MetadataMap</i> . Exactly one of <i>@Format</i> , <i>@Value</i> or the pair <i>@ValueFormat</i> and <i>@ValueTemplate</i> SHALL be specified. Allowed values are from: ▶ Appendix E String Generation.
<i>BarcodeDetails</i> ?	element	Additional specification for complex barcodes.
<i>ExtraValues</i> ?	element	Additional values encoded in the <i>IdentificationField</i> .
<i>MetadataMap</i> *	element	Describes the mapping of metadata that is encoded in an <i>IdentificationField</i> to partition keys. Note: This allows for automated selective finishing based on bar codes.

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 8.21: EncodingDetails Attribute Values (Sheet 1 of 2)

VALUE	DESCRIPTION	VALUE	DESCRIPTION
BOBST		ITF_14	
BrailleASCII	A binary representation for 6 dot Braille messages. See ▶ [Braille ASCII].	ITF_6	
BrailleUnicode	A binary representation for Braille messages. See ▶ [Braille Unicode].	ITF_16	
CODABAR		KURANDT	
CODABAR_Tradional		LAETUS_PHARMA	
CODABLOCK		MSI	
CODABLOCK_F		NDC_HRI	
Code128		PARAF	
Code25		Plessey	
Code39		PDF417	

Table 8.21: EncodingDetails Attribute Values (Sheet 2 of 2)

VALUE	DESCRIPTION	VALUE	DESCRIPTION
Code39_Extended		PZN	
EAN	Includes Bookland_EAN and ISSN.	QR	
EAN_13		RSS_14	
EAN_8		RSS_14_Stacked	
EAN_Coupon		RSS_14_Stacked_Omnidir	
EAN_128		RSS_14_Truncated	
HIBC_Code39		RSS_Limited	
HIBC_Code128		RSS_Expanded	
HIBC_Code39_2		RSS_Expanded_Stacked	
HIBC_CODABLOCK_F		UPC_A	
HIBC_QR		UPC_Coupon	
HIBC_DATAMATRIX		UPC_E	
Interleave25		UPC_SCS	

8.18.1 BarcodeDetails

Table 8.22: BarcodeDetails Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
BarcodeVersion ?	NMTOKEN	The version of a barcode. Values include those from: ▶ Table 8.25 BarcodeVersion Values – for HIBC_DATAMATRIX barcodes. Values include those from: ▶ Table 8.26 BarcodeVersion Values – for QR barcodes.
ErrorCorrectionLevel ?	NMTOKEN	Error correction level for barcodes having a separately definable error correction level. Each value can be used only for certain values of IdentificationField / @EncodingDetails . Values include: PDF417_EC_0 – For @EncodingDetails ="PDF417" PDF417_EC_1 – For @EncodingDetails ="PDF417" PDF417_EC_2 – For @EncodingDetails ="PDF417" PDF417_EC_3 – For @EncodingDetails ="PDF417" PDF417_EC_4 – For @EncodingDetails ="PDF417" PDF417_EC_5 – For @EncodingDetails ="PDF417" PDF417_EC_6 – For @EncodingDetails ="PDF417" PDF417_EC_7 – For @EncodingDetails ="PDF417" PDF417_EC_8 – For @EncodingDetails ="PDF417" QR_EC_L – For @EncodingDetails ="QR" QR_EC_M – For @EncodingDetails ="QR" QR_EC_Q – For @EncodingDetails ="QR" QR_EC_H – For @EncodingDetails ="QR"
XCells ?	integer	The number of cells in the x direction of a matrix barcode. For "DATAMATRIX" this field can be omitted since @BarcodeVersion already defines this. For "PDF417" this is the number of codewords/row.

Table 8.22: BarcodeDetails Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>YCells</i> ?	integer	The number of cells in the y direction of a matrix barcode. For "DATAMATRIX" this field can be omitted since @BarcodeVersion already defines this. For "PDF417" this is the number of rows.

8.18.2 ExtraValues

Table 8.23: ExtraValues Element

NAME	DATA TYPE	DESCRIPTION
<i>Usage</i>	NMTOKEN	The usage of the value. Values include: CompositeCode – This is applicable for barcodes like RSS-14 that have an optional composite code part. Coupon – The additional message for the EAN128 part of a UPC or EAN coupon. Supplemental – UPC supplemental 2/5 digit symbology.
<i>Value</i>	string	Additional value of the IdentificationField as specified in @Usage.

8.18.3 Usage of barcode attributes

The following table specifies whether the **BarcodeReproParams** attributes @Height, @Magnification and @Ratio are applicable for a given barcode type that is specified by @EncodingDetails.

Table 8.24: Usage of Barcode Attributes for Certain Barcode Types (Sheet 1 of 2)

ENCODINGDETAILS VALUES (BARCODE TYPES)	HEIGHT	MAGNIFICATION	RATIO
Code25 Code39 Code39_Extended Interleave25 MSI Plessey	Used	Used	Used
CODABAR Code128 EAN_128 EAN_13 EAN_8 HIBC_Code39 HIBC_Code128 ITF_14 ITF_16 NDC_HRI PARAF UPC_A UPC_E UPC_SCS	Used	Used	Not used
BOBST KURANDT LAETUS_PHARMA	Used	Not used	Not used

Table 8.24: Usage of Barcode Attributes for Certain Barcode Types (Sheet 2 of 2)

ENCODINGDETAILS VALUES (BARCODE TYPES)	HEIGHT	MAGNIFICATION	RATIO
RSS_14 RSS_14_Stacked RSS_14_Stacked_Omnidir RSS_14_Truncated RSS_Limited RSS_Expanded RSS_Expanded_Stacked	Not used	Used	Not used
PZN	Not used	Not used	Not used

The following table specifies valid values of [BarcodeDetails/@BarcodeVersion](#) for an HIBC_DATAMATRIX barcode.

Table 8.25: BarcodeVersion Values – for HIBC_DATAMATRIX barcodes

VALUES			
DM_8_by_18	DM_16_by_16	DM_26_by_26	DM_72_by_72
DM_8_by_32	DM_16_by_36	DM_32_by_32	DM_80_by_80
DM_10_by_10	DM_16_by_48	DM_40_by_40	DM_88_by_88
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

The following table specifies valid values of [BarcodeDetails/@BarcodeVersion](#) for a QR barcode.

Table 8.26: 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 8.2: Barcode

The following example illustrates the description of a barcode in a **LayoutElementProduction** process.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0"
  JobID="LayoutElementProduction" JobPartID="Barcode" Types="LayoutElementProduction">
  <ResourceSet Name="LayoutElementProductionParams" Usage="Input">
    <Resource>
      <LayoutElementProductionParams ContentRefs="Content_000005.1"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Content">
    <Resource ID="Content_000005.1">
      <Content ContentType="Page">
        <BarcodeProductionParams>
          <BarcodeReproParams Height="73.5" Magnification="1">
            <BarcodeCompParams CompensationProcess="Printing" CompensationValue="10"/>
          </BarcodeReproParams>
          <IdentificationField Encoding="Barcode"
            EncodingDetails="EAN_13" Purpose="Label"
            PurposeDetails="ProductIdentification" Value="0123456789128"/>
        </BarcodeProductionParams>
      </Content>
    </Resource>
  </ResourceSet>
</XJDF>
```

8.19 ImageCompression

ImageCompression specifies image compression properties of individual types of images.

Element Properties

Element referenced by: **Content, ImageCompressionParams**

Table 8.27: 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 would otherwise be 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	If "true", the filter defined by <i>@ImageAutoFilterStrategy</i> is applied to photos and the zip compression ("FlateEncode") filter is applied to screen shots. If "false", the <i>@ImageFilter</i> compression method is applied to all images. <i>@AutoFilterImages</i> SHALL NOT be specified unless <i>@EncodeImages</i> is "true". This attribute SHALL NOT be specified if <i>@ImageType</i> = "Monochrome".
<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> ?	float	A value between 0 and 1 that indicates the amount of compression with which the process SHALL compress images when using a "DCTEncode" filter. A value of 0.0 requires that the compression SHOULD be lossless, whereas a value of 1.0 requires 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 <i>@ImageFilter</i> . If "false", no compression filters are applied to sampled images.

Table 8.27: ImageCompression Element (Sheet 2 of 3)

NAME	DATA TYPE	DESCRIPTION
<i>ImageAutoFilterStrategy</i> ?	NMTOKEN	Selects what the image compression strategy to employ if passing through an image that is not already compressed. Values include: JPEG – Lossy JPEG compression for low-frequency images and lossless Flate compression for high-frequency images. JPEG2000 – 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> ?	float	Sets the image downsample threshold for images. This is the ratio of image resolution to output resolution above which downsampling can be performed. For example, if <i>@ImageDownsampleThreshold</i> ="1.5" and <i>@ImageResolution</i> ="72", then 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: Average – The program averages groups of samples to get the new downsampled value. Bicubic – The program uses bicubic interpolation on a group of samples to get a new downsampled value. Subsample – The program picks the middle sample from a group of samples to get the new downsampled value.
<i>ImageFilter</i> ?	NMTOKEN	Specifies the compression filter to be used for images. Ignored if <i>@AutoFilterImages</i> = "true" or if <i>@EncodeImages</i> = "false". Values include: CCITTFaxEncode – Used to select CCITT group 3 or 4 facsimile encoding. SHALL NOT be specified unless <i>@ImageType</i> = "Monochrome". DCTEncode – Used to select JPEG compression. FlateEncode – Used to select zip compression. JBIG2Encode – Used to select JBIG2 encoding. SHALL NOT be specified unless <i>@ImageType</i> ="Monochrome". JPEG2000 – Used to select JPEG2000/Wavelet compression. LZWEncode – Used to select LZW compression. PackBits – Used to select a simple byte-oriented run length scheme.
<i>ImageResolution</i> ?	float	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 whose resolution is above this value.
<i>ImageType</i> ?	enumeration	Specifies the kind of image that SHALL be manipulated. Allowed values are: Color Grayscale Monochrome
<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 Rows and BlackIs1 parameters, which are implicit in the raster data to be compressed.
<i>DCTParams</i> ?	element	The equivalent of the PostScript Columns , Rows and Colors parameters, which are assumed to be implicit in the raster data to be compressed.
<i>FlateParams</i> ?	element	The equivalent of the PostScript Columns , BitsPerComponent and Colors parameters, which are implicit in the raster data to be compressed.
<i>JBIG2Params</i> ?	element	Provides the JBIG2 compression parameters.

Table 8.27: ImageCompression Element (Sheet 3 of 3)

NAME	DATA TYPE	DESCRIPTION
JPEG2000Params ?	element	Provides the JPEG2000 compression parameters.
LZWParams ?	element	The equivalent of the PostScript Columns , BitsPerComponent and Colors parameters, which are implicit in the raster data to be compressed.

8.19.1 CCITTFaxParams

Table 8.28: CCITTFaxParams Element

NAME	DATA TYPE	DESCRIPTION
EncodedByteAlign ?	boolean	@EncodedByteAlign indicates whether the CCITTFaxEncode filter SHALL insert extra 0 bits before each encoded line so that the line begins on a byte boundary.
EndOfBlock ?	boolean	A flag indicating whether the CCITTFaxEncode filter SHALL append an end-of-block pattern to the encoded data.
EndOfLine ?	boolean	A flag indicating whether the CCITTFaxEncode filter SHALL prefix an end-of-line bit pattern to each line of encoded data.
K ?	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 MAY be followed by at most @K – 1 lines encoded two-dimensionally.
Uncompressed ?	boolean	A flag to indicate whether the file generated MAY use uncompressed encoding when advantageous.

8.19.2 DCTParams

Table 8.29: DCTParams Element

NAME	DATA TYPE	DESCRIPTION
ColorTransform ?	enumeration	Color transformation algorithm. Allowed values are: Automatic – "YUV" for 3-channel raster data, "None" otherwise. None – Colors SHALL NOT be transformed. YUV – RGB raster values SHALL be transformed to YUV before encoding and from YUV to RGB after decoding. If four channels are present CMYK values SHALL be transformed to YUVK before encoding and SHALL be transformed from YUVK to CMYK after decoding. Note: YUV is equivalent to YCbCr in TIFF terminology.
HSamples ?	IntegerList	A sequence of horizontal sampling factors. If present, one entry SHALL be provided per color channel in the raster data. If not specified, the implied default is "1" for every channel.
HuffTable ?	FloatList	Huffman tables for DC and AC components. If present, there SHALL be at least one HuffTable element for each color channel.
QFactor ?	float	A scale factor that SHALL be applied to the elements of @QuantTable .
QuantTable ?	FloatList	Quantization tables. If present, there SHALL be one @QuantTable entry for each color channel.
VSamples ?	IntegerList	A sequence of vertical sampling factors. If present, one entry SHALL be provided 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 *ImageCompression* 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.

8.19.3 FlateParams

Table 8.30: FlateParams Element

NAME	DATA TYPE	DESCRIPTION
<i>Predictor</i> ?	integer	A code that selects the predictor function. Note: On "1n" PNG predictors, these values select the specific PNG predictor function(s) to be used. When decoding, the predictor function SHALL be explicitly encoded in the incoming data. Values include: 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 SHALL automatically choose the optimum function separately for each row.

8.19.4 JBIG2Params

Table 8.31: 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.

8.19.5 JPEG2000Params

Table 8.32: JPEG2000Params Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>CodeBlockSize</i> ?	integer	The nominal code block width and height. The value SHALL be a power of 2.
<i>LayerRates</i> ?	FloatList	Compression bit ratio for each layer. If specified, there SHALL be the same number of values in this list as <i>@LayersPerTile</i> in ascending order. Small values correspond to maximum compression and 1.0 corresponds to no compression (lossless). If available, <i>@LayerRates</i> SHOULD be supplied.
<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 that SHALL be encoded in the file.
<i>ProgressionOrder</i> ?	enumeration	Per tile progression order. Allowed values are: CPRL – Component-position-resolution-layer progressive. LRCP – Layer-resolution-component-position progressive (i.e., rate scalable). PCRL – Position-component-resolution-layer progressive. RLCP – Resolution-layer-component-position progressive (i.e., resolution scalable). RPCL – Resolution-position-component-layer progressive.

Table 8.32: JPEG2000Params Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>TileSize</i> ?	XYPair	The width and height of each encoding tile. If not specified the image SHALL be encoded as a single tile.

8.19.6 LZWParams

Table 8.33: LZWParams Element

NAME	DATA TYPE	DESCRIPTION
<i>EarlyChange</i> ?	integer	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.
<i>Predictor</i> ?	integer	A code that selects the predictor function. Note: On "1n" PNG predictors, these values select the specific PNG predictor function(s) to be used. When decoding, the predictor function SHALL be explicitly encoded in the incoming data. Values include: 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 SHALL automatically choose the optimum function separately for each row.

8.20 MediaLayers

MediaLayers contains an ordered list of subelements. Each subelement describes an individual layer of a layered *Media* such as self-adhesive labels or corrugated boards. The first layer in *MediaLayers* SHALL specify the front layer of the *Media* until the last layer, which SHALL define the back. The *Glue* and *Media* subelements MAY therefore be specified in any order and NEED NOT be specified in the lexical order as specified in ▶ Table 8.34 *MediaLayers* Element below.

Element Properties

Element referenced by: *Media*

Table 8.34: MediaLayers Element

NAME	DATA TYPE	DESCRIPTION
<i>Glue</i> *	element	Each <i>Glue</i> SHALL specify a glue layer of a layered <i>Media</i> . The value of <i>Glue/@AreaGlue</i> SHALL be "true".
<i>Media</i> *	element	Each <i>Media</i> SHALL describe an individual layer of a layered <i>Media</i> .

8.21 MetadataMap

MetadataMap allows metadata embedded in PDL files or barcodes that are represented by *IdentificationField* to be assigned to partition key values. If *MetadataMap* is defined in a *RunList*, the metadata SHALL be extracted from the PDL as follows: each *MetadataMap* element SHALL be evaluated for each node (set, document, page, etc.) of the PDL document structure. For XML based PDL files an XPath expression SHALL 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 SHALL be used instead and the value assignment SHALL be performed on the derived XML for the PDL file.

If *MetadataMap* is defined in an *IdentificationField*, then *IdentificationField/@ValueTemplate* SHALL provide a list of variables that can be further processed in *MetadataMap/@ValueTemplate*.

Element Properties

Element referenced by: [IdentificationField](#), [RunList](#)Table 8.35: *MetadataMap* Element

NAME	DATA TYPE	DESCRIPTION
<i>Name</i>	NMTOKEN	<i>@Name</i> SHALL define the partition key that SHALL be filled with the value that is calculated from <i>@ValueFormat</i> and <i>@ValueTemplate</i> . See ▶ Table 6.4 Part Element.
<i>ValueFormat</i>	string	Formatting value for combining values from <i>@ValueTemplate</i> into a dynamic result value. Allowed values are from: ▶ Appendix E String Generation.
<i>ValueTemplate</i>	NMTOKENS	Arguments for combining extracted values. If <i>MetadataMap</i> is a child of <i>RunList</i> , then each value shall be selected from the list of predefined values in ▶ Appendix E String Generation or match a value of <i>Expr/@Name</i> . If <i>MetadataMap</i> is a child of <i>IdentificationField</i> , each value shall be defined in the parent <i>IdentificationField/@ValueTemplate</i> .
<i>Expr</i> *	element	Exactly one <i>Expr</i> element with a matching <i>@Name</i> SHALL be specified for each variable in <i>@ValueTemplate</i> that is NOT defined in the parent <i>IdentificationField/@ValueTemplate</i> and NOT defined in ▶ Table E.1 Predefined variables used in <i>@XXXTemplate</i> . <i>Expr</i> SHALL NOT be specified in an <i>IdentificationField/MetadataMap</i> .

8.21.1 Expr

Expr elements define how the variables that are specified in *@ValueTemplate* SHALL be extracted from the parent *RunList*.Table 8.36: *Expr* Element

NAME	DATA TYPE	DESCRIPTION
<i>Name</i>	NMTOKEN	Name of this <i>Expr</i> . The value extracted from <i>@Path</i> SHALL be used to evaluate the parent <i>@ValueTemplate</i> .
<i>Path</i>	XPath	The value specified by this path SHALL be assigned to <i>Expr/@Name</i> . If the XPath points to an element, then an implied XPath text() function SHALL be executed.

Example 8.3: RunList/MetadataMap

In the following example, the *MetadataMap* element maps the data in */doc/record/Geschlecht* and */doc/record/Status* in the document to *Part/@Metadata*. The calculated *Part/@Metadata* is then used to select the appropriate color and quality of the paper component.

```

<ResourceSet Name="RunList" Usage="Input">
  <Resource>
    <RunList>
      <FileSpec URL="file://host/file/data.pdf"/>
      <MetadataMap Name="Metadata" ValueFormat="%s_%s" ValueTemplate="gender status">
        <Expr Name="gender" Path="/doc/record/Geschlecht"/>
        <Expr Name="status" Path="/doc/record/Status"/>
      </MetadataMap>
    </RunList>
  </Resource>
</ResourceSet>
<ResourceSet Name="Component" Usage="Input">
  <Resource ExternalID="BlueGoodPaper">
    <Part Metadata="Mann_Platin"/>
    <Component/>
  </Resource>
  <Resource ExternalID="BlueCheapPaper">
    <Part Metadata="Mann(.)*/>
    <Component/>
  </Resource>
  <Resource ExternalID="PinkGoodPaper">
    <Part Metadata="Frau_Platin"/>
    <Component/>
  </Resource>
  <Resource ExternalID="PinkCheapPaper">
    <Part Metadata="Frau(.)*/>
    <Component/>
  </Resource>
</ResourceSet>

```

Example 8.4: IdentificationField/MetadataMap

In the following example, barcodes are scanned on a sheet to verify that all sheets have been produced. The three **MetadataMap** elements map the data that is extracted from a barcode to **XJDF/@JobID**, **Part/@DocIndex** and **Part/@SheetIndex**. The first six characters are read into a virtual string variable “job”, which is appended to a fixed string “Job_”, to generate the value for **XJDF/@JobID**. The next three characters are read into a virtual integer variable “doc”, which is used twice to generate a blank separated range value for **Part/@DocIndex**. The final two characters are read into a virtual integer variable “sheet”, which is used twice to generate a blank separated range value for **Part/@SheetIndex**. Thus the barcode “Dec00704216” would generate:

- **XJDF/@JobID="Job_Dec007"**
- **Part/@DocIndex="42 42"**
- **Part/@SheetIndex="16 16"**

```

<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="Barcode"
  JobPartID="Metadata" Types="Verification">
  <ResourceSet Name="Component" Usage="Input">
    <Resource>
      <Component>
        <IdentificationField ValueFormat="%6s%3i%2i" ValueTemplate="job doc sheet">
          <MetadataMap Name="JobID" ValueFormat="Job_%s" ValueTemplate="job"/>
          <MetadataMap Name="DocIndex" ValueFormat="%i %i" ValueTemplate="doc doc"/>
          <MetadataMap Name="SheetIndex" ValueFormat="%i %i" ValueTemplate="sheet sheet"/>
        </IdentificationField>
      </Component>
    </Resource>
  </ResourceSet>
</XJDF>

```

8.22 MISDetails

MISDetails is a container for MIS related information.

Element Properties

Element referenced by: [ResourceInfo](#), [PipeParams](#), [JobPhase](#), [NodeInfo](#)

Table 8.37: MISDetails Element

NAME	DATA TYPE	DESCRIPTION
Complexity ?	float	Complexity of the task specified by this XJDF in a range from 0.0 to 1.0. Note: The interpretation of values is implementation dependent. Values include: 0.0 – The job is simple and therefore reduced setup and waste or higher speeds are possible. 0.5 – The job is of standard complexity and therefore standard setup and waste or normal speeds are possible. 1.0 – The job is complex and therefore more setup and waste or lower speeds are possible.
CostType ?	enumeration	Specifies whether or not this MISDetails is chargeable to the customer or not. Allowed values are: Chargeable NonChargeable
WorkType ?	enumeration	Definition of the work type for this MISDetails (i.e., whether or not this MISDetails relates to originally planned work, an alteration or rework). Allowed values are: Alteration – Work done to accommodate a change made to the job. Original – Standard work that was originally planned for the job. Rework – Work done due to unforeseen problems with the original work (bad plate, resource damaged, etc.).
WorkTypeDetails ?	NMTOKEN	Machine readable definition of the details of the work type for this MISDetails (i.e., why the work was done). Values include: CustomerRequest – The customer requested change(s) requiring the work. EquipmentMalfunction – Equipment used to produce the resource malfunctioned; resource needs to be created again. InternalChange – Change was made for production efficiency or other internal reason. ResourceDamaged – A resource needs to be created again to account for a damaged resource (damaged plate, etc.). UserError – Incorrect operation of equipment or incorrect creation of resource requires creating the resource again.

8.23 Notification

This element contains information about individual events that occurred during processing. For a detailed discussion of event properties, see ▶ Section 9.3.8 Error Handling.

Element Properties

Element referenced by: [AuditNotification](#), [Response](#), [ResponseForceGang](#), [ResponseGangStatus](#), [ResponseKnownDevices](#), [ResponseKnownMessages](#), [ResponseKnownSubscriptions](#), [ResponseModifyQueueEntry](#), [ResponseNotification](#), [SignalNotification](#), [ResponsePipeControl](#), [ResponseQueueStatus](#), [ResponseRequestQueueEntry](#), [ResponseResource](#), [ResponseResubmitQueueEntry](#), [ResponseReturnQueueEntry](#), [ResponseShutdown](#), [ResponseStatus](#), [ResponseStopPersistentChannel](#), [ResponseSubmitQueueEntry](#), [ResponseWakeUp](#)

Table 8.38: Notification Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
Class	enumeration	Class of the notification. Allowed values are from: ▶ Severity.
JobID ?	NMTOKEN	@ JobID that this Notification applies to.

Table 8.38: Notification Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<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>QueueEntryID</i> ?	NMTOKEN	@ <i>QueueEntryID</i> of the QueueEntry during which this Notification was generated.
Comment *	element	A Comment element contains a verbose, human-readable description of the Notification . If multiple Comment elements occur, they SHALL have different Comment / <i>@Language</i> values.
Event ?	element	See Event element below. Not more than one of Event and Milestone SHALL be specified.
Milestone ?	element	See Milestone element below. Not more than one of Event and Milestone SHALL be specified.
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.
<foreign namespace elements> *	element	Any elements in a foreign namespace. Foreign namespace extensions SHOULD NOT duplicate functionality of XJDF .

8.23.1 Event

This element provides additional information for common events.

Table 8.39: Event Element

NAME	DATA TYPE	DESCRIPTION
<i>EventID</i>	NMTOKEN	Internal event ID of the application that emits the event.
<i>EventValue</i> ?	string	Additional user defined value related to this event.

8.23.2 Milestone

In addition to the concrete **XJMF** feedback with respect to process status (see ▶ Section 7.18 Status) and available/consumed resources (see ▶ Section 7.14 Resource), 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: **Milestone** elements usually refer to events involving multiple objects, although **Milestone**/*@MilestoneType* is specified as a singular. The scope of the **Milestone** is defined by the parent **Notification** element.

Table 8.40: Milestone Element

NAME	DATA TYPE	DESCRIPTION
<i>MilestoneType</i>	NMTOKEN	Type of Milestone . Values include those from: ▶ Milestones.
<i>TypeAmount</i> ?	integer	Indication of how many elements have been processed if the milestone refers to certain resources, e.g. the number of pages proofed or the number of different printed sheets. It is not the cumulative amount.

8.24 ObjectResolution

ObjectResolution defines a resolution depending on *@SourceObjects* data types.

Element Properties

Element referenced by: [InterpretingParams](#), [RenderingParams](#)

Table 8.41: ObjectResolution Element

NAME	DATA TYPE	DESCRIPTION
AntiAliasing ?	NMTOKEN	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. Values include: AntiAlias – Anti-aliasing SHALL be applied. The algorithm is system specified. None – Anti-aliasing SHALL NOT be applied.
Resolution	XYPair	Horizontal and vertical output resolution in DPI.
SourceObjects ?	enumerations	Identifies the class(es) of incoming graphical objects to render at the specified resolution. If @SourceObjects is not specified then ObjectResolution SHALL apply to all object classes. Allowed values are from: ▶ SourceObjects .

8.25 OCGControl

OCGControl defines the policy for including or excluding layers that are encoded as ‘Optional Content Groups’ (OCGs) in PDF.

The order of [OCGControl](#) elements SHALL have 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 SHALL define the drawing order of those graphic elements.

Any preferences recorded in an OCG within the PDF file as to whether that OCG SHOULD be displayed or not SHALL be ignored if that OCG is referenced from an [OCGControl](#) element.

The state of all OCGs explicitly referenced from [OCGControl](#) elements SHALL be set before determining the state of any remaining OCGs.

Note: 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.

Element Properties

Element referenced by: [Content](#), [PDFInterpretingParams](#)

Table 8.42: OCGControl Element

NAME	DATA TYPE	DESCRIPTION
IncludeOCG	boolean	Defines whether the optional content group(s) identified by @OCGName are to be included in the RunList . If "true", then the layer SHALL be included. If "false", it SHALL NOT. 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.
OCGName ?	string	The name of the optional content group(s) that SHALL be included or excluded. Exactly one of @OCGName or @ProcStepsGroup SHALL be present. Note: The Name attribute of an optional content group entry is encoded as a PDF text string, and @OCGName is encoded with the Unicode variant identified in the XJDF file header; names SHALL be re-encoded as necessary for comparison.
ProcStepsGroup ?	NMTOKEN	An OCG is selected, if @ProcStepsGroup matches the value of GTS_ProcStepsGroup in the GTS_Metadata dictionary of the OCG of a PDF that complies with ▶ [ISO19593-1:2016].
ProcStepsType ?	NMTOKEN	If specified, an OCG is selected, if @ProcStepsType matches the value of GTS_ProcStepsType in the GTS_Metadata dictionary of the OCG of a PDF that complies with ▶ [ISO19593-1:2016]. @ProcStepsType SHALL NOT be specified unless @ProcStepsGroup is present.

8.26 Perforate

Perforate describes one perforated line.

Element Properties

Element referenced by: [FoldingParams](#), [PerforatingParams](#)

Table 8.43: Perforate Element

NAME	DATA TYPE	DESCRIPTION
<i>Depth</i> ?	float	Depth of the perforation, in microns [μm].
<i>StartPosition</i> ?	XYPair	Starting position of the tool.
<i>TeethPerDimension</i> ?	float	Number of teeth in a given perforation extent in teeth/point. MicroPerforation is defined by specifying a large number of teeth (<i>@TeethPerDimension</i> > 1000).
<i>WorkingDirection</i> ?	enumeration	Direction from which the tool is working. Allowed value is from: ▶ WorkingDirection.
<i>WorkingPath</i> ?	XYPair	Working path of the tool beginning at <i>@StartPosition</i> .

8.27 QueueEntry

The **QueueEntry** element contains metadata for a single item in a device's queue.

Element Properties

Element referenced by: [ResponseModifyQueueEntry](#), [ResponseQueueStatus/Queue](#), [ResponseSubmitQueueEntry](#)

Table 8.44: QueueEntry Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Activation</i> ?	enumeration	Specifies the activation of the QueueEntry . Allowed value is from: ▶ Activation.
<i>EndTime</i> ?	dateTime	Date and time when processing of the QueueEntry 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	Ganging policy for the QueueEntry . Allowed value is from: ▶ GangPolicy.
<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. A value of "0" is the lowest priority, while "100" is the highest priority.
<i>QueueEntryID</i>	NMTOKEN	Identifier of a QueueEntry . This ID SHALL be generated by the queue owner. <i>@QueueEntryID</i> SHALL be unique in the context of a Queue .
<i>StartTime</i> ?	dateTime	Date and time when processing of the QueueEntry has been started.
<i>Status</i>	enumeration	Specifies the status of the requested QueueEntry . <i>@Status</i> SHALL be identical to the <i>NodeInfo/@Status</i> of the underlying XJDF . Allowed value is from: ▶ Status.
<i>StatusDetails</i> ?	NMTOKEN	<i>@StatusDetails</i> provides additional details on the status of the QueueEntry . Values include those from: ▶ Status Details.
<i>SubmissionTime</i> ?	dateTime	Date and time when the entry was submitted to the queue.
<i>FileSpec (Preview)</i> ?	element	This FileSpec MAY be used to provide a visualization of the QueueEntry . FileSpec(Preview) SHOULD reference an image format such as PNG or JPEG.
<i>GangSource</i> *	element	If present, each GangSource SHALL represent the source jobs that are being processed as a gang job by this QueueEntry .

Table 8.44: QueueEntry Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Part</i> *	element	Describes which parts of a job were submitted to the queue. This SHALL be a copy of ResourceSet [@Name="NodeInfo"]/ Resource / Part .

8.28 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.

Element Properties

Element referenced by: [CommandModifyQueueEntry/ModifyQueueEntryParams](#), [QueryQueueStatus/QueueStatusParams](#)

Table 8.45: QueueFilter Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>FirstEntry</i> ?	NMTOKEN	@ QueueEntryID of the first QueueEntry that this QueueFilter applies to. Only QueueEntry elements that are behind this (including this) QueueEntry in the current queue sorting SHALL be selected. If not specified, there is no filtering based on the position of items in the queue.
<i>GangNames</i> ?	NMTOKENS	Gang names of the QueueEntry elements to be returned. If not specified, there is no filtering on QueueEntry / GangName .
<i>JobID</i> ?	NMTOKEN	Return only QueueEntry elements with specified @ JobID . If not specified, there is no filtering on QueueEntry / JobID .
<i>JobPartID</i> ?	NMTOKEN	Return only QueueEntry elements with specified @ JobPartID . If not specified, there is no filtering on QueueEntry / JobPartID .
<i>LastEntry</i> ?	NMTOKEN	@ QueueEntryID of the last QueueEntry that this QueueFilter applies to. Only QueueEntry elements that are in front of this (including this) QueueEntry in the current queue sorting SHALL be selected. If not specified, there is no filtering based on the position of items in the queue.
<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>MaxPriority</i> ?	integer	Only QueueEntry elements with a @ Priority lower than or equal to the value of @ MaxPriority SHALL be provided in the Queue element. If not specified, there is no @ Priority upper bound on candidates.
<i>MinPriority</i> ?	integer	Only QueueEntry elements with a @ Priority higher than or equal to the value of @ MinPriority SHALL be provided in the Queue element. If not specified, there is no @ Priority lower bound on candidates.
<i>NewerThan</i> ?	dateTime	Only QueueEntry elements with a @ SubmissionTime newer than or equal to @ NewerThan SHALL BE provided in the Queue element. If not specified, there is no dateTime upper bound on candidates.
<i>OlderThan</i> ?	dateTime	Only QueueEntry elements with a @ SubmissionTime older than or equal to @ OlderThan SHALL BE provided in the Queue element. If not specified, there is no dateTime lower bound on candidates.
<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 @ Status matching one of the entries in @ StatusList SHALL be returned. If not specified, there is no filtering on QueueEntry / Status . Allowed values are from: ▶ Status .
<i>GangSource</i> *	element	If present only QueueEntry elements that contain a GangSource element that matches at least one of these GangSource elements SHALL be selected. If not specified, there is no filtering based on GangSource .

Table 8.45: QueueFilter Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>Part</i> *	element	Only <i>QueueEntry</i> elements with all specified <i>Part</i> elements SHALL be returned. If not specified, there is no filtering on <i>QueueEntry/Part</i> .

8.29 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: *Content/PositionObj, Layout/StripMark*

Table 8.46: RefAnchor Element

NAME	DATA TYPE	DESCRIPTION
<i>Anchor</i>	enumeration	<i>@Anchor</i> specifies the origin (0,0) of the vector specified in the rotated coordinate system of the related layout element. Allowed value is from: ▶ <i>Anchor</i> .
<i>AnchorType</i>	enumeration	Role of this <i>RefAnchor</i> . Allowed values are: <i>Parent</i> – The layout element referenced by this <i>RefAnchor</i> is a parent. This layout element is transformed with the parent. <i>Sibling</i> – The layout element referenced by this <i>RefAnchor</i> is a sibling. Both layout elements share a common parent. The parent of this layout element SHALL be specified as the <i>RefAnchor</i> of the first child in the chain of siblings.
<i>rRef</i>	IDREF	Reference to a layout element that this layout element is positioned relative to. This shall be one of <i>Layout/@ID, StripMark/@ID</i> or <i>Position/@ID</i> .

8.30 RegisterRibbon

Description of a register ribbon used for book binding. The relationship of visible, hidden and overall length of the register ribbon is shown in ▶ Figure 8-1: RegisterRibbon lengths and coordinate system for BlockPreparation.

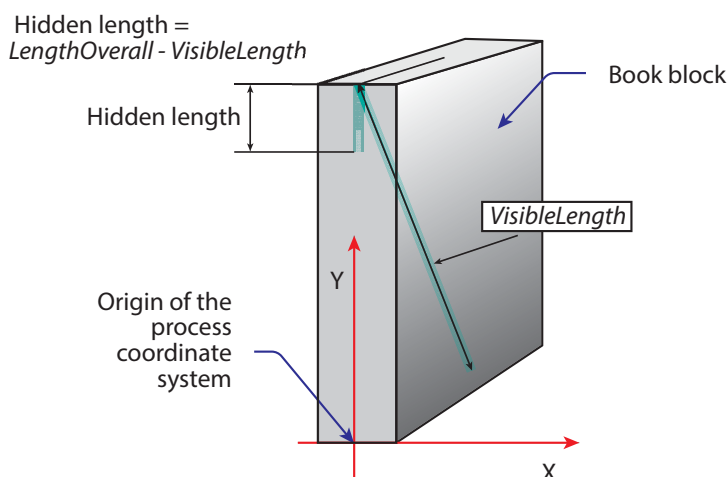
Element Properties

Element referenced by: *BindingIntent/HardCoverBinding, BlockPreparationParams*

Table 8.47: RegisterRibbon Element

NAME	DATA TYPE	DESCRIPTION
<i>LengthOverall</i> ?	float	Overall length of the register ribbon. See ▶ Figure 8-1: RegisterRibbon lengths and coordinate system for BlockPreparation.
<i>Material</i> ?	string	Material of the register ribbon.
<i>RibbonColor</i> ?	enumeration	<i>@RibbonColor</i> specifies the machine readable color of ribbon. Allowed value is from: ▶ <i>NamedColor</i> .
<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.
<i>RibbonEnd</i> ?	NMTOKEN	End of the Ribbon. Values include: <i>Cut</i> <i>CutSealed</i> <i>Knot</i> <i>SealedOffset</i> – The ribbon is sealed some distance from the cut.
<i>VisibleLength</i> ?	float	Length of the register ribbon that will be seen when opening the book. See ▶ Figure 8-1: RegisterRibbon lengths and coordinate system for BlockPreparation.

Figure 8-1: RegisterRibbon lengths and coordinate system for BlockPreparation



8.31 ScreenSelector

Description of screening for a selection of source object types and separations.

Element Properties

Element referenced by: [ColorSpaceConversionOp](#), [Content](#), [ScreeningParams](#)

Table 8.48: ScreenSelector Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
Angle ?	float	Specifies the first angle of the screen when AM screening is used, otherwise @Angle is ignored.
DotSize ?	float	Specifies the dot size of the screen, in microns [μm], when FM screening (@ScreeningType = "FM" or "Adaptive") is used.
Frequency ?	float	Specifies the halftone screen frequency in lines per inch (lpi) of the screen when AM screening is used, otherwise @Frequency is ignored. With some screens, frequency can change as a function of gray level. In this case, the @Frequency value is interpreted for a mid tone (50%) gray level.
ScreeningFamily ?	string	Vendor specific screening family name.
ScreeningType ?	enumeration	General type of screening. Allowed values are: Adaptive AM – Can be line or dot. See @SpotFunction . ErrorDiffusion FM – Includes all stochastic screening types. HybridAM-FM HybridAMline-dot
Separation ?	NMTOKEN	The separation identifier that this ScreenSelector SHALL apply to. Additional details of the colorants SHOULD be provided in ResourceSet [@Name ="Color"].
SourceFrequencyMax ?	float	Specifies the maximum line frequency of screens that SHALL be matched from the source file when screen matching is to be done. Note: This is a filter that selects on which objects to apply this ScreenSelector .
SourceFrequencyMin ?	float	Specifies the minimum line frequency of screens that SHALL be matched from the source file when screen matching is to be done. Note: This is a filter that selects on which objects to apply this ScreenSelector .
SourceObjects ?	enumerations	Identifies the class(es) of incoming graphical objects on which to use the selected screen. If @SourceObjects is not specified then ScreenSelector SHALL apply to all object classes. Allowed values are from: ▶ SourceObjects .

Table 8.48: ScreenSelector Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
SpotFunction ?	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"> CosineDot Cross Diamond Double DoubleDot Ellipse EllipseA EllipseB EllipseC InvertedDouble InvertedDoubleDot InvertedEllipseA InvertedEllipseC InvertedSimpleDot Line LineX LineY Rhomboid Round SimpleDot Square

8.32 Shape

Element Properties

Element referenced by: [ShapeCuttingParams](#), [ShapeDef](#)

Table 8.49: Shape Element

NAME	DATA TYPE	DESCRIPTION
CutBox ?	rectangle	Specification of a rectangular window.
CutOut ?	boolean	If "true", the inside of a specified shape SHALL be removed. If "false", the outside of a specified shape SHALL be removed. An example of an inside shape is a window. An example of an outside shape is a shaped greeting card.
CutPath ?	PDFPath	Specification of a complex path.
DDESCutType ?	integer	<p>Type of cut or perforation used.</p> <p>Values include: a number between "0" and "999" corresponding to a line type as defined in ▶ [DDES3].</p> <p>Note: A value of 101 corresponds to a cut line.</p>
ShapeDepth ?	float	Depth of the shape cut, measured in microns [μm]. If not specified, the shape SHALL be completely cut.
ShapeType	enumeration	<p>Describes any precision cutting other than hole making.</p> <p>Allowed values are:</p> <ul style="list-style-type: none"> Path Rectangular Round RoundedRectangle – Rectangle with rounded corners.
TeethPerDimension ?	float	Number of teeth in a given perforation extent, in teeth/point. MicroPerforation is defined by specifying a large number of teeth ($n > 1000$).

8.33 StripMark

The **StripMark** element specifies automatically generated production marks. Strip marks are generally converted to explicit **MarkObject** elements during the **Stripping** process.

Element Properties

Element referenced by: [Layout](#), [StackingParams](#)/[InsertSheet](#)

Table 8.50: StripMark Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
AbsoluteHeight ?	float	Absolute height of the StripMark in points.
AbsoluteWidth ?	float	Absolute width of the StripMark in points.
Anchor ?	enumeration	Origin of the mark coordinate system. Allowed value is from: ▶ Anchor .
Font ?	NMTOKEN	The name of the font that SHALL be used for the StripMark . Values include: Courier Helvetica Helvetica-Condensed Times-Roman
FontSize ?	float	The size of the font that SHALL be used for the StripMark , in points.
HorizontalFitPolicy ?	enumeration	How to modify the mark to fit into the horizontal space that is provided for the StripMark prior to rotation. Allowed value is from: ▶ FitPolicy .
ID ?	ID	Identifier of the StripMark . Used for internal references within the Layout .
MarkName ?	NMTOKEN	Name of the mark that SHALL be marked on the Layout . Values include those from: ▶ Table 8.51 MarkName Attribute Values.
Offset ?	XYPair	Position of the anchor of this StripMark relative to RefAnchor / @Anchor as defined by @Anchor and RefAnchor / @Anchor .
Orientation ?	enumeration	Orientation of this StripMark in the coordinate system of the related object defined by RefAnchor . Allowed value is from: ▶ Orientation .
RelativeHeight ?	float	Height relative to the height of the related object defined by RefAnchor of this StripMark .
RelativeWidth ?	float	Width relative to the width of the related object defined by RefAnchor of this StripMark .
StripMarkDetails ?	string	More detailed information about the StripMark . For example, if @MarkName ="Set" or @MarkName ="TabMark" then @StripMarkDetails is a name to refer to a proprietary or site specific set of marks.
VerticalFitPolicy ?	enumeration	Policy of how to modify the mark to fit into the vertical space that is provided for the StripMark prior to rotation. Allowed value is from: ▶ FitPolicy .
BarcodeReproParams ?	element	Description of the formatting and reproduction parameters for dynamically generated barcodes.
FillMark *	element	Specifies marks that define a fill layer, e.g., for backlit displays.
IdentificationField *	element	Contents of barcodes.
JobField ?	element	Details of automatically generated text marks. JobField SHOULD NOT be specified unless @MarkName ="JobField", @MarkName ="TabMark" or @MarkName ="WaterMark".

Table 8.50: StripMark Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>RefAnchor</i>	element	Reference to an element that defines the coordinate system that this mark SHALL be placed relative to.

Figure 8-2: Anchor with no scaling and rotation of 90° clockwise

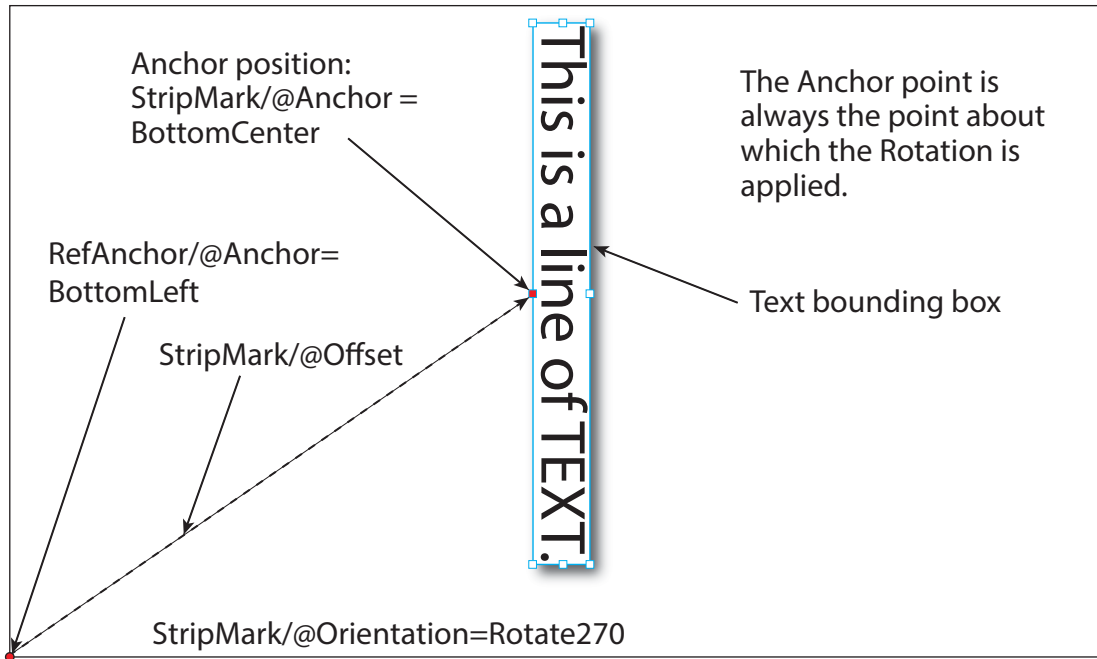


Table 8.51: MarkName Attribute Values (Sheet 1 of 2)

VALUE	DESCRIPTION
BleedMark	
CenterMark	
CIELABMeasuringField	
CollationMark	
ColorControlStrip	
ColorRegisterMark	
CutMark	
DensityMeasuringField	
FillMark	Background fill (e.g., for backlit display).
FoldMark	
GrommetMark	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.
IdentificationField	
JobField	
PaperPathRegisterMark	

Table 8.51: MarkName Attribute Values (Sheet 2 of 2)

VALUE	DESCRIPTION
RegisterMark	
ScavengerArea	
Set	Specifies to use a MarkSet (file containing multiple marks). The name of the MarkSet MAY be passed in @StripMarkDetails .
TabMark	Specifies automatically generated data for tab blocks.
TrimMark	
WaterMark	A faint design imaged onto the surface during the printing process typically for protection and imaging as a lighter background to text or images.

8.33.1 FillMark

Table 8.52: FillMark Element

NAME	DATA TYPE	DESCRIPTION
KnockoutBleed ?	float	Bleed in points that the fill SHALL grow into (positive values) from the knockout area. Note: This attribute implies the same bleed for all separations.
KnockoutRefs ?	IDREFS	Reference to the PlacedObject , Position or StripMark elements that SHALL not be filled by this FillMark . The knockout boundaries are defined by the value of @KnockoutSource .
KnockoutSource	enumeration	Definition of the source of the knockout from the referenced PlacedObject elements. Allowed values are: ClipPath – Use the clip path as defined by the referenced PlacedObject / @ClipPath . SourceClipPath – Use the clip path as defined by the referenced PlacedObject / @SourceClipPath . TrimBox – Use the clip path as defined by the referenced PlacedObject / @TrimCTM and PlacedObject / @TrimSize .
MarkColor +	element	Definition of the separations used to fill the mark.

8.33.2 MarkColor

Definition of the separations used to fill a dynamic mark.

Table 8.53: MarkColor Element

NAME	DATA TYPE	DESCRIPTION
Name	string	Identifier of the separation. Additional details SHOULD be specified in ResourceSet [@Name = "Color"].
Tint	float	Value from 0 (not used) to 1 (100% tint) of the separation specified in @Name .

8.33.3 JobField

A [JobField](#) is a mark object that specifies the details of a job. [JobField](#) elements are also referred to as slug lines.

Table 8.54: JobField Element (Sheet 1 of 2)

NAME	DATA TYPE	DESCRIPTION
JobFormat ?	string	A formatting string used with @JobTemplate to generate a string. Allowed values are from: ▶ Appendix E String Generation.

Table 8.54: JobField Element (Sheet 2 of 2)

NAME	DATA TYPE	DESCRIPTION
<i>JobTemplate</i> ?	NMTOKENS	A list of values used with @ <i>JobFormat</i> to generate a string. Allowed values are from: ▶ Appendix E String Generation.

8.34 SubscriptionInfo

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

Element Properties

Element referenced by: *ResponseKnownSubscriptions*, *SignalKnownSubscriptions*, *ResponseStopPersistentChannel*

Table 8.55: SubscriptionInfo Element

NAME	DATA TYPE	DESCRIPTION
<i>ChannelID</i>	NMTOKEN	@ <i>ChannelID</i> specifies the <i>Header</i> / <i>@ID</i> of the <i>Query</i> message that initiated the <i>Subscription</i> . @ <i>ChannelID</i> SHALL match <i>Header</i> / <i>@refID</i> of each <i>Signal</i> that is transmitted on this persistent channel.
<i>DeviceID</i> ?	NMTOKEN	Identifier of the controller that subscribed for the persistent channel. @ <i>DeviceID</i> SHALL match <i>Header</i> / <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 <i>Signals</i> that comprise this persistent channel.
<i>Subscription</i>	element	The <i>Subscription</i> element that describes the persistent channel.

9 Building a System

A ▶ Device SHALL be able to consume the inputs and produce the outputs for each process type it is able to execute.

9.1 Queue Support

In **XJMF**, a controller or device is assumed to have one input queue that accepts and manages queue entries by responding to **CommandSubmitQueueEntry**, **CommandResubmitQueueEntry** and **CommandModifyQueueEntry**. Queue entries SHALL be returned to the submitting controller using a **CommandReturnQueueEntry**. 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 that supports pending jobs can be mapped to a queue with either no **QueueEntry** elements or with one **QueueEntry** element where **@Status="InProgress"**. **XJMF** supports simple handling of priority queues. The following assumptions are made:

- Queues MAY support priority.
- Priority SHALL only be changed if **QueueEntry/@Status="Waiting"** or **QueueEntry/@Status="Held"**.
- 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 after the last pending priority 100 job.
- Queue entries SHALL be unambiguously identified by **QueueEntry/@QueueEntryID**.
- A controller or device MAY analyze an **XJDF** that is submitted to its queue either at submission or at 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.

9.1.1 Queue Entry ID Generation

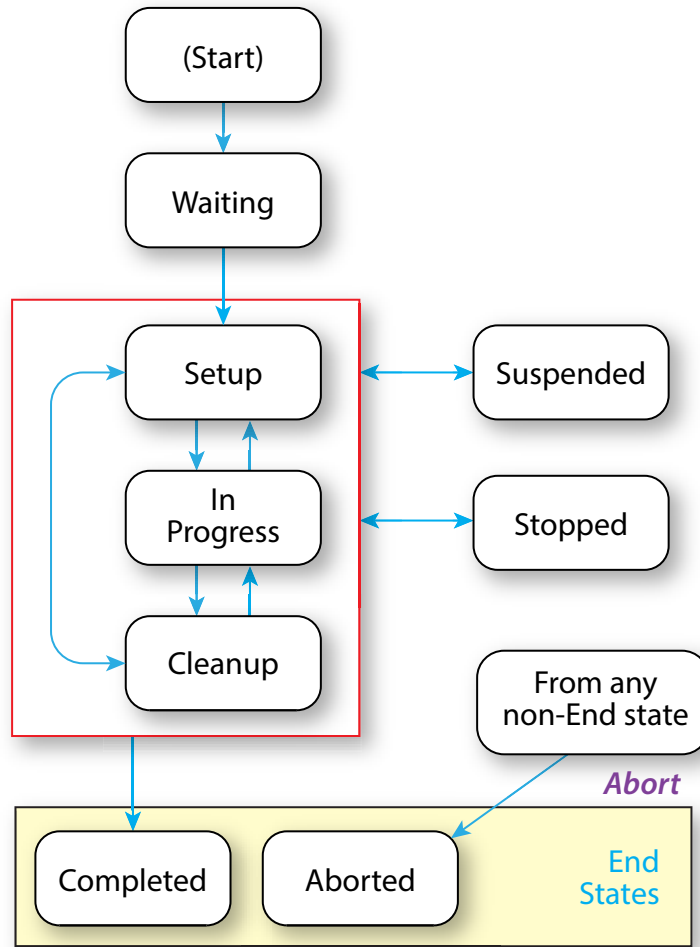
Queue entries are identified by the **QueueEntry/@QueueEntryID** attribute, which the queue’s device SHALL generate when it receives and accepts the submitted job, and which SHALL be returned in the **ResponseSubmitQueueEntry**. **@QueueEntryID** SHALL uniquely identify an entry within the scope of one queue. An implementation is free to choose the algorithm that generates **@QueueEntryID** values.

9.2 Status Transitions

A process that is represented by an **XJDF** will go through various states during its life time as described in ▶ Figure 9-1: Life Cycle of a process and queue entry. These states are defined in detail in **NodeInfo/@Status**.

Note: The process or queue entry 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 9-1: Life Cycle of a process and queue entry MAY be skipped and processing SHALL continue with the next supported phase.

Figure 9-1: Life Cycle of a process and queue entry



9.3 Execution Model

The processing model of **XJDF** is based on a producer/consumer model. Devices that process **XJDF** act both as producers and consumers of resources.

9.3.1 Determining Executable XJDF

In order to determine which parts of an **XJDF** can be executed, the controller or device SHALL use the following procedures.

- 1 The controller or device SHOULD select one or more partitions with **NodeInfo/@Status="Waiting"** taking into account the scheduling attributes in **NodeInfo**.
- 2 The controller or device SHOULD determine if no **ResourceSet[@Usage="Input"]/Resource/@Status="Unavailable"** for the selected partition.

9.3.2 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** with the same **@JobID** to the same or a different device.

9.3.3 Partial Processing of XJDF with Partitioned ResourceSets

Some processes apply to multiple parts such as multiple sheets or plates. The structure of **ResourceSet** [**@Name="NodeInfo"**] SHALL define the sequencing of the process steps. The device SHALL process all work steps that are explicitly specified in **NodeInfo Resources**.

If a device is only capable of performing more granular worksteps than the **NodeInfo** partition structure 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.

9.3.4 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 imposition 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** with the same **@JobID** to a different device or to the same device if it is capable of processing multiple work steps simultaneously, e.g. a multi-threaded raster image processor.

9.3.5 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 **CommandPipeControl** messages. If this is technically not feasible, out of band communication such as a pallet of printed paper that was delivered by a fork lift **MAY** be used as process control.

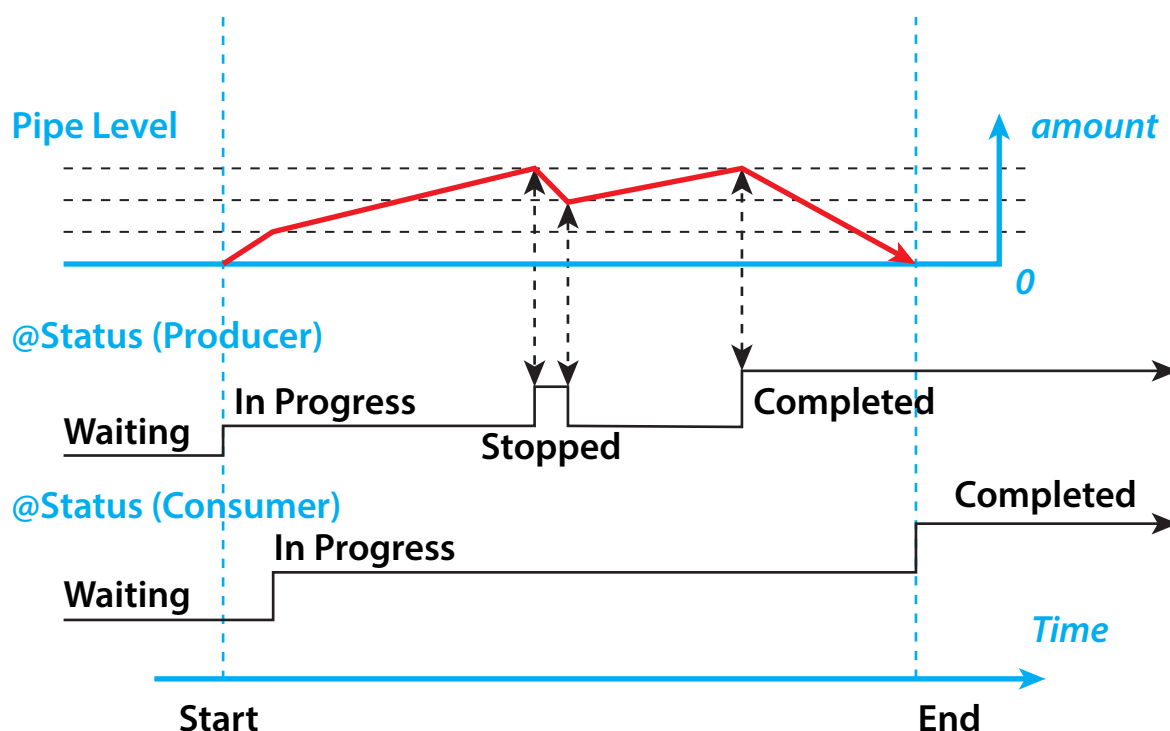
ResourceSet/Dependent provides information to setup a pipe. Any **ResourceSet** **MAY** be defined as a pipe resource by specifying **Dependent/@PipeProtocol**.

The following figure illustrates how the level of a pipe resource rises and falls in relationship to the **NodeInfo/@Status** of the producing and consuming processes.

Dependent/@XJMFURL SHALL specify the URL that receives **CommandPipeControl XJMF** messages.

Dependent/@PipeProtocol SHALL specify the protocol used to control the pipe.

Figure 9-2: Example of process status and resource amount in overlapping processing



9.3.5.1 Dynamic Pipes

In addition to abstractly declaring pipe properties, **XJMF** provides the **CommandPipeControl** message that allows 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.

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 **CommandPipeControl** messages.

Dependent/[@XJMFURL](#) specifies the recipient of **CommandPipeControl** messages. Depending on the values of the **Dependent**/[@PipeProtocol](#) attribute, the following actions are possible.

"XJMFPull":

The consumer initiates the pipe by sending a **CommandPipeControl**/[PipeParams](#)/[@Operation="Pull"](#) message to its [@XJMFURL](#). The consumer MAY request new resources by sending **CommandPipeControl**/[PipeParams](#)/[@Operation="Pull"](#) messages. If the producer is incapable of fulfilling **CommandPipeControl**/[PipeParams](#)/[@Operation="Pull"](#) messages for other reasons (such as a malfunction), it SHOULD send a **CommandPipeControl**/[PipeParams](#)/[@Operation="Pause"](#) message to the consumer. Once the producer is again capable of supplying (e.g. the malfunction has been removed), it SHOULD send a **CommandPipeControl**/[PipeParams](#)/[@Operation="Push"](#) message to the consumer to inform the consumer that it can commence sending [@Operation="Pull"](#) messages. The consumer SHOULD send a **CommandPipeControl**/[PipeParams](#)/[@Operation="Close"](#) message to the producer if the consumer does not require any further resources.

"XJMFPush":

The producer initiates the pipe by sending a **CommandPipeControl**/[PipeParams](#)/[@Operation="Push"](#) message to its [@XJMFURL](#). The producer MAY dispatch new resources by sending **CommandPipeControl**/[PipeParams](#)/[@Operation="Push"](#) messages. If the consumer is incapable of fulfilling **CommandPipeControl**/[PipeParams](#)/[@Operation="Push"](#) requests for other reasons (such as a malfunction), it SHOULD send a **CommandPipeControl**/[PipeParams](#)/[@Operation="Pause"](#) message to the producer. Once it is again ready to consume resources (e.g. the malfunction has been removed), it SHOULD send a **CommandPipeControl**/[PipeParams](#)/[@Operation="Pull"](#) to inform the producer that it can commence sending **CommandPipeControl**/[PipeParams](#)/[@Operation="Push"](#) messages. The producer SHOULD send a **CommandPipeControl**/[PipeParams](#)/[@Operation="Close"](#) message to the consumer if the producer cannot provide any further resources.

Dynamic pipes are initially dormant and SHALL be activated by an explicit request. If **Dependent**/[@PipeProtocol="XJMF"](#), dynamic pipe requests MAY be initiated by either end of the pipe. As soon as the pipe has been initiated, actions that are required by the implied [@PipeProtocol](#) ("XJMFPush" or "XJMFPull") SHALL be applied. For example, a print process might notify an off-line finishing process when a certain amount is ready by sending a **CommandPipeControl**/[PipeParams](#)/[@Operation="Push"](#) message, or the printing process might request a new plate by sending a **CommandPipeControl**/[PipeParams](#)/[@Operation="Pull"](#) message.

9.3.5.2 Comparison of Non-Dynamic and Dynamic Pipes

Each **Dependent** element between non-dynamic pipes provides the pipe definitions for the process to which the **Dependent** element belongs. Therefore, many processes can link to the same pipe to enable parallel processing.

In contrast, dynamic pipes provide a URL address to control a process. 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 pallet at a time, it is up to the implementation to ensure consistency of the processes.

9.3.5.3 Metadata in Pipe Messages

PipeParams/[ResourceSet](#) can contain metadata that is required by the recipient of the message. This metadata SHALL be specified as partition keys in [ResourceSet](#)/[Resource](#)/[Part](#) and additional details MAY be specified as the actual contents of the [ResourceSet](#). Partition key metadata provides a mechanism to retain context in large variable data jobs without requiring completely expanded [ResourceSets](#) with potentially thousands of [Resource](#) elements in the **XJDF**.

A typical example of partition key metadata is [Part](#)/[@DocIndex](#), [Part](#)/[@RunIndex](#) and [Part](#)/[@Side](#) to uniquely identify the context of a surface image that is sent from a RIP to a digital press.

9.3.6 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 using an [ApprovalDetails](#) that MAY be specified as an input [ResourceSet](#) in any process.

The **Approval**, **QualityControl** and **Verification** processes accept any [ResourceSet](#) as input. These processes output a [ResourceSet](#) of the same type as the input [ResourceSet](#) and an [ApprovalDetails](#), [QualityControlResult](#) or [VerificationResult](#) [ResourceSet](#). For hard copy proofing, a **DigitalPrinting** process generates the hard proof that is input to an **Approval** process. For soft proofing, a **Rendering** or **PDLCreation** process generates the soft proof that 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 [ResourceSet](#).

9.3.7 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. **CommandForceGang** allows gang to be released to a device and [QueryGangStatus](#) pro-

vides information about the currently known gangs. For a description of planned job ganging, see also ▶ Section 5.4.21 SheetOptimizing.

9.3.8 Error Handling

Error handling is an implementation-dependent feature of **XJDF** based systems. **AuditPool** provides a container where errors that occur during the execution of an **XJDF** SHOULD be logged as **AuditNotification** elements. **Notification** elements MAY also be sent in **XJMF SignalNotification** messages. The content of the **Notification** element is described in ▶ Table 8.38 Notification Element. For a list of predefined error codes, see ▶ Appendix B Return Values.

9.3.8.1 Classification of Notifications

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 **Notification/@Class** in ▶ Table 8.38 Notification Element.

9.3.8.2 Event Description

A description of the event SHOULD be given in the **Notification/Comment** element, which SHALL be specified for the **Notification** with **@Class="Information"**, **"Warning"**, **"Error"** or **"Fatal"**. For example, after a process is aborted, error information describing a device error SHOULD be logged in the **Comment** element of the **Notification** element.

9.3.8.3 Error Handling via Messaging (XJMF)

An **XJMF** with a **SignalNotification** message 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 9.6.3 Managing Persistent Channels.

9.4 Specifying Complex Processing

There are occasions where a controller might need to provide details of multiple individual processes to a controller such as a prepress workflow system or production control system in the context of an individual job. This can be achieved by submitting an initial **XJDF** with **SubmitQueueEntry** and submitting the individual process **XJDF** with a **ResubmitQueueEntry** as follows:

The controller SHALL submit an **XJDF** with a new **XJDF/@JobID**. This **XJDF** SHOULD have a value of **XJDF/@Types** that contains **"Product"** and SHOULD provide an **XJDF/ProductList** that completely describes the desired products.

Additional processes SHALL be supplied by sending one or more **ResubmitQueueEntry XJMF**. These messages SHALL reference a process **XJDF** where the value of **XJDF/@JobID** is identical to the primary **XJDF/@JobID** and the value of **XJDF/@JobPartID** is different from the value of the primary **XJDF/@JobPartID**.

XJDF/ResourceSet specifies the respective resource in the context of the submitted process **XJDF**. **ResourceSet** elements SHALL be identified by **ResourceSet/@ID**. Thus two **ResourceSet** elements in two **XJDF** elements with the same **ResourceSet/@ID** represent the same physical objects. **ResourceSet/@ID** NEED NOT be maintained over multiple **XJDF** instances. **XJDF/ResourceSet/Dependent** elements MAY be specified to explicitly setup process dependencies.

9.4.1 Referencing Multiple XJDF in a Directory

If **QueueSubmissionParams/@URL** of the original **XJDF** references a directory, then all contained files with an extension of **".xjdf"** SHALL be processed in lexically sorted ascending order. The first entry is processed as a logical **SubmitQueueEntry** and the second and further entries are processed as logical **ResubmitQueueEntry** commands.

The first two digits of the file names of the **XJDF** files in the directory SHOULD begin with a numerical character, i.e. a character in the range '0' to '9' in order to ensure a well defined lexical ordering.

9.5 XJDF and XJMF Interchange Protocol

XJDF and **XJMF** SHOULD be exchanged over a network by using http ▶ [RFC2616] or https.

Controllers and devices SHOULD provide insecure http without a TLS 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.

9.5.1 HTTP Port

XJMF messaging does not specify a standard port.

9.5.2 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 ▶ Section 9.7 XJDF 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.

9.5.3 HTTPS-Based Protocol – TLS

Secure **XJMF** has no additional requirements in addition to standard TLS ▶ [RFC5246].

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.

9.6 XJMF Handshaking

This section describes the actions and appropriate reactions in a communication between controllers and devices using **XJMF**.

9.6.1 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. *Header/@refID* SHALL be set to the value of *Header/@ID* of the message from the sender. If the incoming message could not be parsed, the response SHALL be a **ResponseNotification**.

9.6.1.1 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 B Return Values and that SHOULD contain a **Notification** element that SHOULD provide additional details of the error.

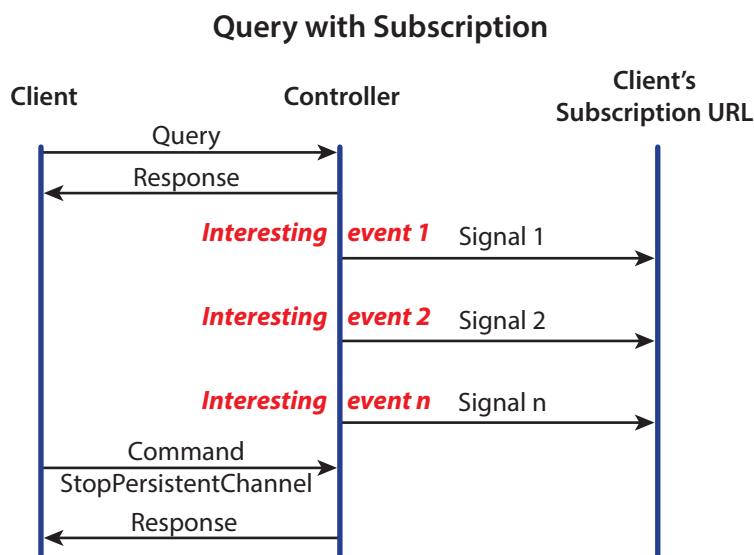
The response messages contain a *@ReturnCode* attribute. *@ReturnCode* defaults to 0, which indicates that the response is successful. In case of success and in responses to commands an informational **Notification** element (*@Class="Information"*) MAY be provided. In case of a warning, error or fatal error, the *@ReturnCode* is greater than 0 and indicates the kind of error that occurred. In this case, a **Notification** element SHOULD be provided. Error codes are defined in ▶ Appendix B Return Values.

Example 9.1: Response with Notification Element

The following example uses a **Notification** element to describe an error:

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="DeviceID" ID="1_000002" Time="2018-02-28T16:00:17+00:00"/>
  <ResponseModifyQueueEntry ReturnCode="5">
    <Header DeviceID="DeviceID" ID="R1" Time="2018-02-28T16:00:17+00:00" refID="C1"/>
    <Notification Class="Error">
      <Comment>StartJob unsuccessful - Device does not handle resume</Comment>
    </Notification>
  </ResponseModifyQueueEntry>
</XJMF>
```

Figure 9-3: Interaction of messages with a subscription



9.6.2 Subscribing for Signals

Queries SHALL be subscribed to by including a **Subscription** element that defines the details of the subscription. The receiver of the subscription SHALL initially send a response message containing only **@ReturnCode** and any appropriate **Notification** elements 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 and set the **@ReturnCode** to "111".

The following examples illustrate the subscription handshake for **SignalStatus** including the first signal.

Example 9.2: Status Subscription

The following **QueryStatus** subscription requests a time trigger **SignalStatus** every thirty seconds.

```

<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:17+00:00"/>
  <QueryStatus>
    <Header DeviceID="TestSender" ID="Status1" Time="2018-02-28T17:00:00+00:00"/>
    <Subscription RepeatTime="30" URL="http://MIS:1234/xjmfurl"/>
    <StatusQuParams/>
  </QueryStatus>
</XJMF>
  
```

Example 9.3: Status Subscription Response

The following **ResponseStatus** to the subscription above is empty and **@ReturnCode="0"** defines success.

```

<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:21+00:00"/>
  <ResponseStatus ReturnCode="0">
    <Header DeviceID="TestSender" ID="1_000003"
      Time="2018-02-28T17:00:00+00:00" refID="Status1"/>
  </ResponseStatus>
</XJMF>
  
```

Example 9.4: Status Subscription Signal

The following **SignalStatus** is a simple signal status by a device that is currently producing.

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="DeviceID" ID="1_000002" Time="2018-02-28T16:00:19+00:00"/>
  <SignalStatus>
    <Header DeviceID="DeviceID" ID="S1" Time="2018-02-28T17:00:00+00:00" refID="Status1"/>
    <DeviceInfo Status="Production">
      <JobPhase JobID="j1" JobPartID="p1"
        StartTime="2018-02-28T17:00:00+00:00" Status="InProgress"/>
    </DeviceInfo>
  </SignalStatus>
</XJMF>
```

9.6.3 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.

Persistent channels SHOULD be maintained, even when a device is powered off and powered on again.

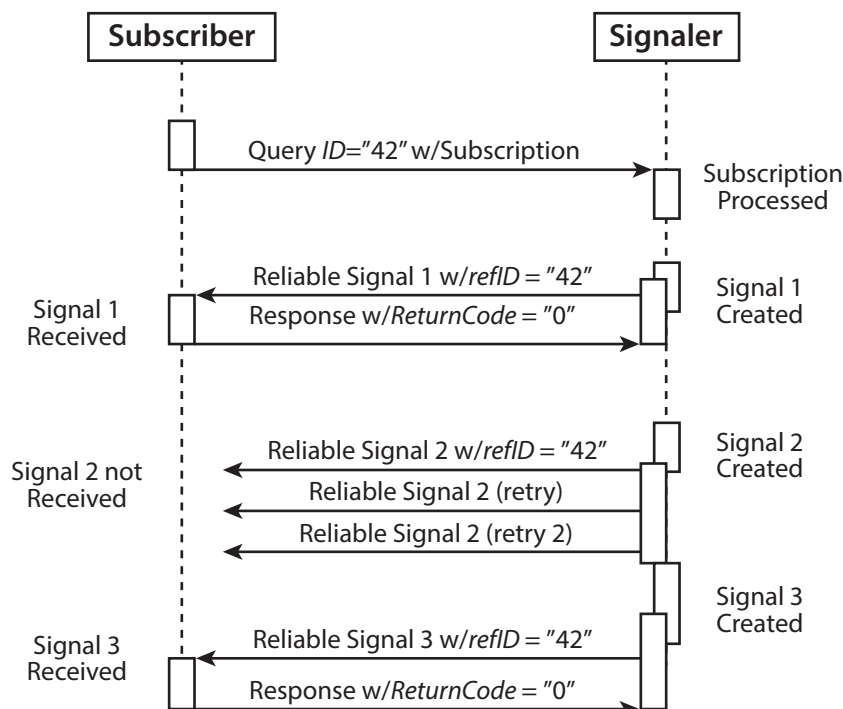
9.6.4 Signal Handshaking

JMF signal messages that were subscribed with **Subscription/@ChannelMode="FireAndForget"** SHALL NOT be resent in case they were not successfully delivered to the signal subscriber. In case of success, the subscriber 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 9.6.1.1 XJMF Error Handling.

9.6.5 Reliable Signaling

JMF signal messages that were subscribed with **Subscription/@ChannelMode="Reliable"** SHALL be resent in case they were not successfully delivered to the signal subscriber. If the receiver does not respond to the reliable signal, the sender SHALL retry the reliable signal. 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. For instance, a heartbeat signal that is triggered by **Subscription/@RepeatTime** MAY be replaced by the following signal if no information that is required for job costing is lost.

Figure 9-4: Example of reliable signaling



9.6.5.1 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.

9.6.6 Deleting Persistent Channels

A persistent channel SHALL be deleted by sending a [StopPersistentChannel](#) command message.

9.6.7 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 below:

Known message discovery

- 1a Controller sends [QueryKnownMessages](#) to device.
The controller SHOULD query for known messages and refrain from sending unknown messages, including messages specified in this section.
- 1b Device sends [ResponseKnownMessages](#) to controller.
The device SHOULD respond with a list of known messages.

Device discovery

- 2a Controller sends [QueryKnownDevices](#) to device.
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](#).
- 2b Device sends [ResponseKnownDevices](#) to controller.
The device SHOULD respond with a list of known lower-level devices.

Subscription discovery

- 3a Controller sends [QueryKnownSubscriptions](#) to device.
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.
- 3b Device sends [ResponseKnownSubscriptions](#) to controller.
The device SHOULD respond with a list of known subscriptions.

9.6.8 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 an **XJDF**.

9.7 XJDF Packaging

An **XJMF** MAY be transferred with no additional packaging. Alternatively, an **XJMF** and its referenced digital 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.

XJMF messages that do not refer to **XJDF** or external digital assets SHOULD NOT be packaged as zip. Digital assets that are not included in the zip package MAY be referenced. Multiple **XJMF** messages SHALL NOT be packaged in one zip package.

9.7.1 MIME Types and File Extensions

The following MIME types and extensions SHOULD be used when storing **XJDF** or **XJMF** as files or when a MIME type is required, e.g. when setting the http Content-Type header.

Table 9.1: MIME Types and File Extensions

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	xjdf.zip	Zip packaged XJDF . A double extension of .xjdf.zip SHOULD be used.
application/vnd.cip4-xjmf+zip	xjmf.zip	Zip packaged XJMF . A double extension of .xjmf.zip SHOULD be used.

9.7.2 ZIP Packaging

Zip is a de facto industry standard for packaging and compressing data. Directory structures can be encoded in a zip package. For details see ▶ [ZIP].

9.7.2.1 Identifying the Root XJMF

The root **XJMF** SHALL be named root.xjmf and SHALL reside in the root directory of the zip package.

9.7.2.2 Referencing Digital Assets within a ZIP Package

Referenced digital 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 **XJDF** within the zip package.

Digital assets other than the root.xjmf MAY be placed in a directory tree structure within the zip file.

9.7.2.3 ZIP File Name Encoding

All file and directory names in a zip package SHALL be encoded in UTF-8.

Note: Zip allows any encoding but provides no method to declare the encoding.

9.8 Job 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.

9.8.1 Rescheduling with ModifyQueueEntry

ModifyQueueEntry is designed to allow rescheduling of jobs without changing any parameters of the **XJDF**. 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.

9.8.2 Modifying Jobs

ResubmitQueueEntry is designed to modify the details of the underlying sets of **XJDF** for a job. Jobs can be modified in a number of different ways. These are differentiated by **ResubmissionParams/@UpdateMethod** and the related **XJDF/@JobPartID**.

Table 9.2: Modifying Job Parameters

RESUBMISSION-PARAMS/ @UPDATE-METHOD	XJDF/ @JOBPARTID	DESCRIPTION
Complete	-	If XJDF/@JobPartID is omitted, the job parameters of all XJDFs that belong to the queue entry SHALL be completely overwritten with new information.
Complete	known	If the value of XJDF/@JobPartID matches an existing job part, then the job parameters relating to @JobPartID SHALL be completely overwritten with new information.
Remove	known	If the value of XJDF/@JobPartID matches an existing job part, then 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 @JobPartID SHALL be removed.
Incremental	known	If the value of XJDF/@JobPartID matches an existing job part, then the job parameters that relate to the existing XJDF are overwritten by the data that is explicitly supplied in the referenced XJDF .
Incremental	new	If XJDF/@JobPartID is supplied and does not match any existing job part, then a new process step is requested. Details SHALL be supplied in the referenced XJDF .

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.

9.8.2.1 Referencing values for incremental update

The following sections describe 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.

9.8.2.1.1 Finding the correct XJDF to update

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

9.8.2.1.2 Finding the correct Resource to update

A **ResourceSet** SHALL match if the values of **ResourceSet/@Name**, **ResourceSet/@ProcessUsage** and **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** elements is not significant.

9.8.2.2 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.

9.8.2.3 Removing values

If **ResubmissionParams/@UpdateMethod="Remove"** then all resources that are found according to ▶ Section 9.8.2.1.2 Finding the correct Resource to update SHALL be removed. If a **ResourceSet** with no child resource elements is provided, then the entire **ResourceSet** SHALL be removed.

The following example of an **XJMF** with a referenced **XJDF** removes the **VarnishingParams Resource** for the sheet with **@SheetName="Body"**:

```
<XJMF xmlns="http://www.CIP4.org/JDFSchema_2_0">
  <Header DeviceID="TestSender" ID="1_000002" Time="2018-02-28T16:00:21+00:00"/>
  <CommandResubmitQueueEntry>
    <Header DeviceID="TestSender" ID="C1" Time="2018-02-28T16:00:21+00:00"/>
    <ResubmissionParams QueueEntryID="qe1"
      URL="http://jobserver.xjdf.org?job1" UpdateMethod="Remove"/>
  </CommandResubmitQueueEntry>
</XJMF>
```

The above example references the following **XJDF**.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="ChangeOrder"
  JobPartID="RemoveVarnish" Types="Varnishing">
  <ResourceSet Name="NodeInfo" Usage="Input">
    <Resource>
      <Part SheetName="Body"/>
      <NodeInfo/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="VarnishingParams">
    <Resource>
      <Part SheetName="Body"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

9.8.3 Examples for Job Modification

The following examples are valid but simplified examples of change orders. Real life examples will typically contain additional details.

9.8.3.1 Rescheduling

The following example reschedules the planned start of a process.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="ChangeOrder"
  JobPartID="Reschedule" Types="Folding">
  <ResourceSet Name="NodeInfo" Usage="Input">
    <Resource>
      <Part SheetName="Sheet1"/>
      <NodeInfo Start="2018-02-28T17:00:00+00:00"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

9.8.3.2 Changing Amount

The following example updates the requested amount of Sheet1 to 4000.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="ChangeOrder"
  JobPartID="Amount" Types="Folding">
  <ResourceSet Name="Component" Usage="Output">
    <Resource>
      <AmountPool>
        <PartAmount Amount="4000"/>
      </AmountPool>
      <Part SheetName="Sheet1"/>
    </Resource>
  </ResourceSet>
</XJDF>
```


9.8.3.3 Adding a color separation

The following example adds a color separation "Acme ColorBook 42" to sheets with @SheetName="Sheet1".

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="ChangeOrder"
  JobPartID="AddColor" Types="ConventionalPrinting">
  <ResourceSet Name="NodeInfo" Usage="Input">
    <Resource>
      <Part SheetName="Sheet1"/>
      <NodeInfo/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Media">
    <Resource ID="Media_000004.1">
      <Media MediaType="Plate"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="ExposedMedia" Usage="Input">
    <Resource>
      <Part Separation="Spot1" SheetName="Sheet1" Side="Front"/>
      <ExposedMedia MediaRef="Media_000004.1"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Color" Usage="Input">
    <Resource>
      <Part Separation="Spot1"/>
      <Color ActualColorName="Acme ColorBook 42" CMYK="0.2 0.3 0.4 0.1"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Ink" Usage="Input">
    <Resource Brand="Acme Ink 42">
      <Part Separation="Spot1"/>
      <Ink InkType="Ink"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="ColorantControl" Usage="Input">
    <Resource>
      <Part SheetName="Sheet1" Side="Front"/>
      <ColorantControl ColorantOrder="Cyan Magenta Yellow Black Spot1" ColorantParams="Cyan Magenta
Yellow Black Spot1"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

9.8.3.4 Selecting a Device

The following example changes the device to Folder2.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="ChangeOrder"
  JobPartID="Device" Types="Folding">
  <ResourceSet Name="NodeInfo" Usage="Input">
    <Resource>
      <Part SheetName="Sheet1"/>
      <NodeInfo Start="2018-02-28T17:00:00+00:00"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Device" Usage="Input">
    <Resource>
      <Part SheetName="Sheet1"/>
      <Device DeviceID="Folder2"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

9.8.3.5 Modifying the selected paper

The following example changes the paper weight to 120 g/m² for Sheet1.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="ChangeOrder"
  JobPartID="Paper" Types="ConventionalPrinting">
  <ResourceSet Name="Component" Usage="Input">
    <Resource>
      <AmountPool>
        <PartAmount Amount="4000"/>
      </AmountPool>
      <Part SheetName="Sheet1"/>
      <Component MediaRef="Media_000005.1"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Media">
    <Resource ID="Media_000005.1">
      <Media MediaType="Paper" Weight="120"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

9.8.3.6 Adding an offline varnishing step

The following example adds a varnishing step including the description of the blanket and varnish.

```
<XJDF xmlns="http://www.CIP4.org/JDFSchema_2_0" JobID="ChangeOrder"
  JobPartID="AddVarnish" Types="Varnishing">
  <ResourceSet Name="NodeInfo" Usage="Input">
    <Resource>
      <Part SheetName="Sheet1"/>
      <NodeInfo/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Media">
    <Resource ID="Media_000004.1">
      <Media MediaType="Blanket"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="ExposedMedia" Usage="Input">
    <Resource>
      <Part Separation="Var" SheetName="Sheet1" Side="Front"/>
      <ExposedMedia MediaRef="Media_000004.1"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Color" Usage="Input">
    <Resource>
      <Part Separation="Varnish"/>
      <Color ActualColorName="Acme Gloss Varnish"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Ink" Usage="Input">
    <Resource Brand="Acme Gloss Varnish">
      <Part Separation="Varnish"/>
      <Ink InkType="Gloss Varnish"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="VarnishingParams" Usage="Input">
    <Resource>
      <Part SheetName="Sheet1" Side="Front"/>
      <VarnishingParams VarnishMethod="Blanket"/>
    </Resource>
  </ResourceSet>
  <ResourceSet Name="Device" Usage="Input">
    <Resource DescriptiveName="Offline Acme varnisher">
      <Part SheetName="Sheet1" Side="Front"/>
      <Device DeviceID="Var_1"/>
    </Resource>
  </ResourceSet>
</XJDF>
```

9.9 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's schema for **XJDF** provides functionality to define the supported individual values of any attributes in XML. It currently does not provide conditional constraints for values that depend on the value of other attributes within the **XJDF**. This limitation is seen as acceptable for the benefit of using standard XML tools and may be addressed in a future version of this specification.

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

A Data Types and Values

This appendix lists the **XJDF** data types and describes how they are encoded in XML. The appendix also contains commonly used closed list enumerations, e.g. ▶ Activation, and preferred values for open lists of NMTOKEN or strings, e.g. ▶ Contact Types.

All of the **XJDF** types are derived from XML schema types that are defined in "XML Schema Part 2 - Datatypes", see ▶ [XMLSchema], either by extension, use of lists, or by restriction.

A.1 XJDF Data Types

All **XJDF** data types are described in the following table.

Table A.1: XJDF Data Types (Sheet 1 of 3)

XJDF DATATYPE	XML SCHEMA DATATYPE	DERIVATION	DESCRIPTION
boolean	xsd:boolean	Restriction	String literals only are permitted. Numeric values "0" and "1" SHALL NOT be used.
CMYKColor	xsd:float	Restricted list	The list SHALL contain four values, each in the range of [0...1.0]. A value of 0.0 specifies no ink and a value of 1.0 specifies full ink. The sequence of colors SHALL be "C M Y K".
dateTime	xsd:dateTime	None	Represents a specific instant of time. It shall be a UTC time or a local time that includes the time zone.
duration	xsd:duration	None	Represents a duration of time.
enumeration	xsd:NMTOKEN	Restriction	Individual token from a closed set of values.
enumerations	xsd:NMTOKENS	Restricted list	List of one or more unique tokens from a closed set of values.
float	xsd:float	None	Corresponds to the ▶ [IEEE754] single-precision, 32-bit floating point type. For details, see ▶ [XMLSchema].
FloatList	xsd:float	List	A list of float values.
hexBinary	xsd:hexBinary	None	Represents arbitrary hex encoded binary data.
ID	xsd:ID	None	Unique identifier as defined by ▶ [XML]. SHALL be unique within the scope of the XJDF document.
IDREF	xsd:IDREF	None	Reference to an element holding the unique identifier as defined by [XML Specification 1.0].
IDREFS	xsd:IDREFS	None	List of references (IDREF values) separated by white spaces as defined by ▶ [XML].
integer	xsd:integer	None	Represents numerical integer values. Values greater than +/- 2**31 are not expected to occur for this data type. For details, see ▶ [XMLSchema].
IntegerList	xsd:integer	List	A list of integer values.

Table A.1: XJDF Data Types (Sheet 2 of 3)

XJDF DATATYPE	XML SCHEMA DATATYPE	DERIVATION	DESCRIPTION
IntegerRange	xsd:integer	Restricted list	The list SHALL contain two values representing a range of values. An IntegerRange MAY be used to select a contiguous set of items from a list, as defined in ▶ Section 1.9.2 Counting in XJDF. In this case the two values represent an inclusive range of index values to be selected from the target list, e.g. "a b" selects items $I_a \dots I_b$ inclusive and "m m" the single item I_m .
LabColor	xsd:float	Restricted list	The list SHALL contain exactly three values in the sequence "L a b". The value of L is restricted to a range of [0..100]; a and b are unbounded. Note: Values 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 015:2004] and ▶ [ISO13655:2017]. This corresponds to a white point of X = 0.9642, Y = 1.0000 and Z = 0.8249 in CIEXYZ color space.
language	xsd:language	None	Represents a language and country code (for example, en-US) for a natural language. Values SHALL conform to ▶ [RFC3066].
languages	xsd:language	List	A list of language values.
matrix	xsd:float	Restricted list	The list SHALL contain six values representing the sequence "a b c d Tx Ty". The variables Tx and Ty describe distances, which are defined in points. For more details see ▶ Section 2.6.3 Coordinate System Transformations.
NMTOKEN	xsd:NMTOKEN	None	A continuous sequence of special characters as defined by ▶ [XML]. Note: NMTOKEN values MAY begin with any non whitespace character, including numerical characters.
NMTOKENS	xsd:NMTOKENS	None	Whitespace-separated list of NMTOKEN values.
PDFPath	xsd:string	Restriction	Values of type PDFPath are encoded as a string that conforms to a sequence of PDF path operators. Note: PDF operators are limited to those described in "Path Construction Operators" in ▶ [PDF1.6].
rectangle	xsd:float	Restricted list	The list SHALL contain four values representing "llx lly urx ury".
regExp	xsd:normalizedString	None	Regular expression as defined by ▶ [XMLSchema].
RGBColor	xsd:float	Restricted list	The list SHALL contain three values representing the sequence "r g b". A value of 0.0 SHALL specify no intensity (black) and a value of 1.0 SHALL specify full intensity.
shape	xsd:float	Restricted list	The list SHALL contain three values representing the sequence "width height depth" that are the same as "x y z".
string	xsd:normalizedString	Restriction	The length of the string SHALL NOT exceed 1023 characters. In order to enable fixed length storage of strings in databases, string values SHALL NOT be longer than 1023 characters. Note: Tabs, linefeeds etc. are not valid characters.
text	xsd:string	None	String data in the body of an XML element. Note: This is the only data type that is not encoded as an XML attribute.

Table A.1: XJDF Data Types (Sheet 3 of 3)

XJDF DATATYPE	XML SCHEMA DATATYPE	DERIVATION	DESCRIPTION
TransferFunction	xsd:float	Restricted list	The list SHALL contain an even number of values representing a sequence of "x ₀ y ₀ x ₁ y ₁ ... x _n y _n " pairs. See ▶ Section A.1.1 TransferFunction.
URI	xsd:anyURI	None	Values of type URI represent a Uniform Resource Identifier, as defined in ▶ [RFC3986]. Note: The URI data type is represented as an Internationalized Resource Identifier (IRI) as defined in ▶ [RFC3987].
URL	xsd:anyURI	None	Values of type URL represent a Uniform Resource Locator, as defined in ▶ [RFC3986]. Note: The URL data type is represented as an Internationalized Resource Identifier (IRI) as defined in ▶ [RFC3987]. Note: 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.
XPath	xsd:token	None	Note: Values of type XPath represent an XPath expression as described in ▶ [XPath].
XYPair	xsd:float	Restricted list	The list SHALL contain two values representing the sequence "x y".

A.1.1 TransferFunction

Values 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 < = first stimuli, then the result is the y value of the first xy pair.
- 2 If x > = 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.

A.2 Enumerations

This section contains tables each with a closed set of values for an enumeration or enumerations type. If there are any implications to the order of the values this will be detailed in the description, otherwise no order is implied.

A.2.1 Action

Action specifies what action if any to take as a result of a particular event.

Table A.2: Action Enumeration Values

VALUE	DESCRIPTION
Abort	Abort the ongoing activity and do not proceed with any other further activity.
Continue	Continue with the present activity. Details SHOULD be logged.
Repair	Repair the condition before proceeding with the activity. Details SHOULD be logged. Note: The actions required to perform the repair are system specified.

A.2.2 Activation

Activation SHALL specify the activation of a [QueueEntry](#).

Note: The values in the following table are ordered from least active to most active.

Table A.3: Activation Enumeration Values

VALUE	DESCRIPTION
Informative	The QueueEntry is for information only. If a QueueEntry is "Informative", it SHALL NOT be processed. Queue entries with @Activation = "Informative" will generally be sent to an operator console for preview but are still completely under the control of an external controller.
Held	The QueueEntry has been held and SHALL NOT be processed until its @Activation is changed to "Active". The transition to "Active" MAY be triggered with a CommandModifyQueueEntry/ModifyQueueEntryParams/@Operation="Resume" .
Active	The QueueEntry is active and SHALL be processed regularly.
PendingReturn	Indicates that the QueueEntry has been processed but has not yet been successfully returned to the respective controller.
Removed	The QueueEntry has been removed. This @Activation SHALL NOT be provided unless Queue/@UpdateGranularity = "ChangesOnly" .

A.2.3 Anchor

Anchor specifies the nine anchor points of a rectangle.

Table A.4: Anchor Enumeration Values (Sheet 1 of 2)

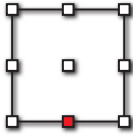
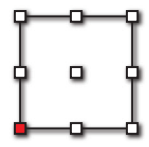
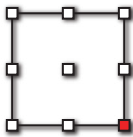
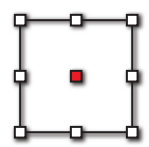
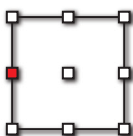
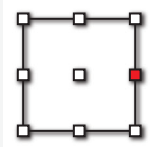
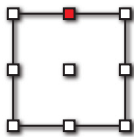
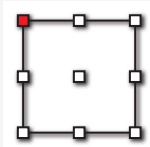
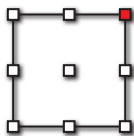
VALUE	DESCRIPTION
BottomCenter	
BottomLeft	
BottomRight	
Center	
CenterLeft	

Table A.4: Anchor Enumeration Values (Sheet 2 of 2)

VALUE	DESCRIPTION
CenterRight	
TopCenter	
TopLeft	
TopRight	

A.2.4 Automation

Automation specifies how complete an item is.

Table A.5: Automation Enumeration Values

VALUE	DESCRIPTION
Dynamic	The item is incomplete and should be completed automatically.
Static	The item is complete.

A.2.5 Axis

Axis specifies the notional line around which an operation, such as mirroring, SHALL be performed.

Table A.6: Axis Enumeration Values

VALUE	DESCRIPTION
Both	The operation is performed around both axes.
FeedDirection	The operation is performed around the feed direction axis.
MediaWidth	The operation is performed around the media width axis.
None	No operation is to be performed.

A.2.6 BinderMaterial

BinderMaterial specifies the material that SHALL be used for loose binding.

Table A.7: BinderMaterial Enumeration Values (Sheet 1 of 2)

VALUES	DESCRIPTION
ColorCoatedSteel	Coated steel.

Table A.7: BinderMaterial Enumeration Values (Sheet 2 of 2)

VALUES	DESCRIPTION
Plastic	Any kind of plastic.
Steel	Plain steel.

A.2.7 BindingType

BindingType specifies the required style of binding to be used.

Table A.8: BindingType Enumeration Values

VALUE	DESCRIPTION
AdhesiveNote	Binding with removable adhesive on the back side of a product. Typically used for small brightly colored paper designed to be stuck prominently to an object or surface and easily removed when necessary.
ChannelBinding	Metal clamps are used to bind sheets. This type of binding is handled by the LooseBinding process.
CoilBinding	Metal wire, plastic coated wire or pure plastic wire is used to fasten pre-punched sheets of paper, cardboard or other materials. This type of binding is handled by the LooseBinding process.
CombBinding	Plastic insert wraps through pre-punched holes in the substrate. This type of binding is handled by the LooseBinding process.
CornerStitch	Stitch in the corner that is at the clockwise end of the binding edge. This type of binding is handled by the Stitching process.
EdgeGluing	Gluing gathered sheets at one edge of the pile. This type of binding is handled by the Gluing process. Products of this type are also referred to as padded.
HardCover	This type of binding defines a hardcover bound book. This type of binding is handled by the CaseMaking process.
LooseBinding	Generic loose binding – one of "ChannelBinding", "CoilBinding", "CombBinding", "RingBinding" or "StripBinding". These types of binding are handled by the LooseBinding process.
None	This type of binding defines a stack of pages with no additional binding.
RingBinding	Pre-punched sheets are placed in a ring binder. This type of binding is handled by the LooseBinding process.
SaddleStitch	Sheets are bound together using stitches along the middle fold which is on a saddle. This type of binding is handled by the Stitching process.
SideStitch	Sheets are bound together using stitches along the reference edge. This type of binding is handled by the Stitching process.
SoftCover	This type of binding defines a softcover bound book. It includes perfect binding and is handled by the CoverApplication process.
StripBinding	Hard plastic strips are held together by plastic pins, which in turn are bound to the strips with heat. This type of binding is handled by the LooseBinding process.
Tape	This type of binding is an inexpensive version of "SoftCover". It is handled by the CoverApplication process.
WireComb	Wire is used to fasten pre-punched sheets of paper, cardboard or other materials. This type of binding is handled by the LooseBinding process.

A.2.8 BundleType

BundleType specifies the type of items that are bundled.

Table A.9: BundleType Enumeration Values

VALUES	DESCRIPTION
BoundSet	Stack of components that are bound together.
Box	Convenience packaging that is not envisioned to be protection for shipping.
Carton	Protection packaging typically used for shipping.
CollectedStack	Components collected on a saddle, e.g. as a result of the Collecting process.
CompensatedStack	Loose stack of compensated components.
Pallet	
Product	An individual product.
Roll	Rolled components on a print roll.
Sheet	Multiple individual items printed on one sheet.
Stack	Loose stack of equally stacked components.
StrappedCompensatedStack	Strapped stack of compensated components.
StrappedStack	Strapped stack of equally stacked components.
WrappedBundle	

A.2.9 ChannelMode

ChannelMode specifies the reliability mode of a message channel.

Table A.10: ChannelMode Enumeration Values

VALUES	DESCRIPTION
FireAndForget	The receiver of the signal MAY respond using an XJMF response message.
Reliable	Indicates that the signal is the result of a subscription where reliable signaling was specified. The receiver of the signal SHALL respond using an XJMF response message.

A.2.10 Coating

Coating specifies the coating of a substrate.

Table A.11: Coating Enumeration Values

VALUE	DESCRIPTION
Coated	A coating of a system specified type.
Gloss	A glossy coating.
Matte	A matte coating.
None	No coating.
Satin	A coating between "Gloss" and "Matte".

A.2.11 Compensation

Compensation specifies how a process SHALL apply transfer curve compensation.

Table A.12: Compensation Enumeration Values

VALUES	DESCRIPTION
Film	Compensated until film exposure.
None	No compensation.
Plate	Compensated until plate exposure.
Press	Compensated until press.

A.2.12 DataType

DataType is used to specify the data type of a value where it cannot be inferred from the context and thus needs to be explicitly stated. It is therefore expected that DataType will be suitably paired with an item containing the value.

Table A.13: DataType Enumeration Values

VALUES	DESCRIPTION
boolean	Binary value logic, either "true" or "false".
dateTime	Represents a specific instant of time. It SHALL be a UTC time or a local time that includes the time zone.
duration	Represents a duration of time.
float	Corresponds to the ▶ [IEEE754] single-precision, 32-bit floating point type. For details, see ▶ [XMLSchema].
integer	Represents numerical integer values. For details, see ▶ [XMLSchema].
NamedFeature	This represents a ▶ NamedFeature as defined in ▶ Section 3.1.3.1 Specifying NamedFeatures with GeneralID . Note: NamedFeature describes a value that is identified by a specific name, thus in this case it is expected to have both an item containing the value and an item containing the name. For example: <pre><GeneralID DataType="NamedFeature" IDUsage="pool" IDValue="bar snax" /></pre>
NMTOKEN	A continuous sequence of special characters as defined by ▶ [XML].
string	Character strings without tabs or line feeds. Corresponds to the standard XML normalizedString data type. For details, see ▶ [XMLSchema].

A.2.13 DeviceStatus

DeviceStatus specifies the state of a device.

Table A.14: DeviceStatus Enumeration Values

VALUE	DESCRIPTION
Idle	No job is being processed and the device is accepting new jobs.
NonProductive	The device is not doing productive work but rather doing something like maintenance or running a test job.
Offline	The device is either switched off, cannot be accessed or is in stand-by that requires a wake-up.
Production	At least one job is in a productive status on the device, i.e. XJDF/@Status="Setup", "Running" or "Cleanup" .
Stopped	At least one job is in a productive status on the device, i.e. XJDF/@Status="Setup", "Running" or "Cleanup" , and the device has been stopped and requires attention. This status indicates some kind of break as long as execution has not been aborted.

A.2.14 Drying

Drying specifies the method employed to dry an item.

Table A.15: Drying Enumeration Values

VALUE	DESCRIPTION
Heatset	Heatset dryer.
IR	Infrared dryer.
Off	No dryer is used.
On	The device's default drying unit is used.
UV	Ultraviolet dryer.

A.2.15 Edge

Edge specifies the edge of an object.

Table A.16: Edge Enumeration Values

VALUES	DESCRIPTION
Bottom	Bottom edge of a sheet or product.
Left	Left edge of a sheet or product.
Right	Right edge of a sheet or product.
Top	Top edge of a sheet or product.

A.2.16 EmbossDirection

EmbossDirection specifies the type and direction of embossing.

Table A.17: EmbossDirection Enumeration Values

VALUES	DESCRIPTION
Both	Both debossing and embossing using one stamp.
Depressed	Debossing only.
Flat	The embossing foil is applied flat. Used for foil stamping.
Raised	Embossing only.

A.2.17 EmbossType

EmbossType specifies the type of embossing required.

Table A.18: EmbossType Enumeration Values

VALUES	DESCRIPTION
BlindEmbossing	Embossed forms are not inked or foiled. The color of the image is the same as the substrate.
Braille	Six dot braille embossing.
EmbossedFinish	The overall design or pattern is impressed in laminated paper when passed between metal rollers engraved with the desired pattern. It is produced on a special embossing device to create finishes such as linen.
FoilEmbossing	Combines embossing and foil stamping in a single operation.
FoilStamping	Uses a heated die to place a metallic or pigmented image from coated foil onto the substrate.

A.2.18 Face

Face specifies the location on a three dimensional object, e.g. [Component](#).

Table A.19: Face Enumeration Values

VALUE	DESCRIPTION
Back	Back side of a sheet or product.
Bottom	Bottom of a product.
Front	Front side of a sheet or product.
Left	Left side of a product, e.g. the spine of a left bound book.
Right	Right side of a product, e.g. the spine of a right bound book.
Top	Top of a product.

A.2.19 FeedQuality

FeedQuality specifies the action of a feeder in response to a feeder failure condition.

Table A.20: FeedQuality Enumeration Values

VALUES	DESCRIPTION
Check	Check the quality and register.
NotActive	Quality control is not active.
StopNoWaste	Check the quality and register. The consuming device SHALL stop after the predefined number of consecutive errors. The error SHALL be corrected, e.g. manually.
StopWaste	Check the quality and register. The object failing the test SHALL be waste. The consuming device SHALL stop after the predefined number of consecutive errors. The error SHALL be corrected, e.g. manually.
Waste	The object failing the test SHALL be waste.

A.2.20 FitPolicy

FitPolicy specifies how an object should be manipulated to enable it to fit into a given area.

Note: The 'given direction' in the following text is derived from the attribute's context, i.e. for [@HorizontalFitPolicy](#) this would be horizontal.

Table A.21: FitPolicy Enumeration Values

VALUES	DESCRIPTION
NoRepeat	The object is neither resized nor repeated. If it is bigger than the given area then it SHALL be clipped.
RepeatToFill	The object SHALL be placed in the requested position. It SHALL then be repeated in the given direction, allowing clipping to occur, until all the allocated space is filled.
RepeatUnclipped	The object SHALL be placed in the requested position. It SHALL then be repeated in the given direction, without clipping, to fill as much of the allocated space as possible.
StretchToFit	The object SHALL be stretched along the given direction to entirely fill the allocated space. Note: If used in isolation this can result in distortion of the object's aspect ratio.
UndistortedScaleToFit	The object SHALL be resized to fit in the given direction. Note: For the orthogonal direction this may result in either the object being clipped or the object not filling the allocated space.

A.2.21 GangPolicy

GangPolicy specifies how multiple jobs SHALL be ganged.

Table A.22: GangPolicy Enumeration Values

VALUES	DESCRIPTION
Gang	The job SHALL be ganged and MAY be submitted to the device.
GangAndForce	The job SHALL be ganged and SHALL be submitted to the device.
NoGang	The job SHALL NOT be ganged.

A.2.22 Glue

Glue specifies the type of glue to be used.

Table A.23: Glue Enumeration Values

VALUES	DESCRIPTION
ColdGlue	
Hotmelt	
PUR	Polyurethane rubber.

A.2.23 IncludeResources

IncludeResources specifies how fonts SHALL be embedded.

Table A.24: IncludeResources Enumeration Values

VALUES	DESCRIPTION
IncludeNever	Never embed fonts.
IncludeOncePerDoc	Embed once per document.
IncludeOncePerPage	Embed once per page.

A.2.24 ISOPaperSubstrate

ISOPaperSubstrate specifies a print substrate according to ▶ [ISO12647-2:2013].

Note: See ▶ Section C.3 Paper Grade for a mapping to the paper grade values defined in ▶ [ISO12647-2:2004].

Table A.25: ISOPaperSubstrate Enumeration Values

VALUE	DESCRIPTION
PS1	Premium coated.
PS2	Improved coated.
PS3	Standard coated glossy.
PS4	Standard Coated Matte.
PS5	Wood-free Uncoated.
PS6	Super Calendered.
PS7	Improved Uncoated.
PS8	Standard Uncoated.

A.2.25 MappingSelection

MappingSelection specifies how a device should construct a color.

Table A.26: MappingSelection Enumeration Values

VALUE	DESCRIPTION
UseLocalPrinterValues	Use the device's best local mapping.
UsePDLValues	Use color values specified in the PDL. See ▶ [ColorPS].
UseProcessColorValues	Use the values defined in the associated process.

A.2.26 MediaDirection

MediaDirection specifies a preferred orientation of a characteristic of **Media** such as grain or flute.

Table A.27: MediaDirection Enumeration Values

VALUE	DESCRIPTION
Any	No restrictions apply to alignment of the media property.
SameDirection	The media property SHALL be aligned along the same axis of the coordinate system for all items.
XDirection	The media property SHALL be aligned along the X-axis of the coordinate system.
YDirection	The media property SHALL be aligned along the Y-axis of the coordinate system.

A.2.27 MediaType

MediaType specifies the general type of media to be used.

Table A.28: MediaType Enumeration Values (Sheet 1 of 2)

VALUE	DESCRIPTIONS
Blanket	A blanket used for varnishing.
CorrugatedBoard	
Disc	CD or DVD disc to be printed on.
EmbossingFoil	
Film	
Foil	
GravureCylinder	Gravure cylinder.
ImagingCylinder	Reusable direct imaging cylinder in a press.
LaminatingFoil	
MountingTape	For flexo platem mounting tape. New in JDF 1.4
Other	Something other than a media defined by this table.
Paper	Unprinted paper. Includes single layer cardboard.
Plate	
Screen	Used for screen printing.
SelfAdhesive	
ShrinkFoil	
Sleeve	Flexo sleeve.

Table A.28: MediaType Enumeration Values (Sheet 2 of 2)

VALUE	DESCRIPTIONS
Textile	
Transparency	
Vinyl	

A.2.28 NamedColor

NamedColor specifies a machine readable definition of a color. For a list of allowed values see ▶ [Color Names].

A.2.29 Opacity

Opacity specifies the opacity of a resource.

Table A.29: Opacity Enumeration Values

VALUE	DESCRIPTION
Opaque	The media or resource is opaque and does not transmit light under normal incident lighting conditions.
Translucent	The media or resource is translucent. For example, translucent material can be used for back lit viewing.
Transparent	The media or resource is transparent.

A.2.30 Orientation

Orientation specifies the orientation of a **Resource**. For details see ▶ Table 2.1 Matrices and Orientation values for describing the orientation of a Component.

Table A.30: Orientation Enumeration Values

VALUE	EQUIVALENT TRANSFORMATION MATRIX	DESCRIPTION
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
Flip270	0 1 1 0 0 0	270° Counterclockwise Rotation + Flip around X

Note: In the transformation matrix above, ‘h’ and ‘w’ refer to the height and width of the object being transformed.

A.2.31 Polarity

Polarity specifies whether a given image SHALL be color inverted.

Table A.31: Polarity Enumeration Values

VALUE	DESCRIPTION
Negative	The image is color-inverted.
Positive	The image is not color-inverted.

A.2.32 PositionPolicy

PositionPolicy specifies the level of freedom when applying placement or positioning values.

Table A.32: PositionPolicy Enumeration Values

VALUE	DESCRIPTION
Exact	The values SHALL be followed precisely.
Free	The values are used as guidance and MAY be modified by the designer.

A.2.33 RenderingIntent

RenderingIntent specifies the rendering intent that SHALL be applied when rendering the selected object. Values are defined in ▶ [ICC.1].

Table A.33: RenderingIntent Enumeration Values

VALUE	DESCRIPTION
AbsoluteColorimetric	
ColorSpaceDependent	The rendering intent is dependent on the color space. The dependencies are implementation specific.
Perceptual	
RelativeColorimetric	
Saturation	

A.2.34 Scope

Scope specifies the availability of resources and amounts in a device.

Table A.34: Scope Enumeration Values

VALUES	DESCRIPTION
Allowed	The resources are potentially available but currently not available without operator intervention.
Estimate	The amount of resources is an estimate that a device has calculated within the scope of a job.
Job	The amount of resources is an actual measurement of data that is currently available within the scope of a job.
Present	The resources are currently available without operator intervention.

A.2.35 Severity

Severity specifies the severity of an error.

Note: This table is not ordered alphabetically - it is ordered by increasing level of severity.

Table A.35: Severity Enumeration Values

VALUES	DESCRIPTION
Event	Normal operating event.
Information	Informational event worthy of being logged.
Warning	A minor error. The executing device is able to repair the condition and continue.
Error	A significant error. Operator intervention is required to allow the device to continue.
Fatal	A fatal error. The device has aborted the operation and cannot continue.

A.2.36 SheetLay

SheetLay specifies the reference edge where media or components are placed in a **Device**. SheetLay SHALL be specified in the **Device** coordinate system and therefore applies to the media or component after any rotation specified in **Resource/@Orientation** or **Resource/@Transformation** has been applied.

Table A.36: SheetLay Enumeration Values

VALUE	DESCRIPTION
Center	The media is placed in the center. This is most commonly used in web devices.
Left	The media is placed so that it is guided on the left.
Right	The media is placed so that it is guided on the right.

A.2.37 Side

Side specifies which side is to be used for an action.

Table A.37: Side Enumeration Values

VALUES	DESCRIPTION
Back	The back surface.
Front	The front surface.

A.2.38 Sides

Sides specifies the sides of the media or product that SHALL be imaged.

Table A.38: Sides Enumeration Values

VALUE	DESCRIPTION
OneSided	Page contents SHALL be imposed on the front side.
OneSidedBack	Page contents SHALL be imposed on the back side.
TwoSidedHeadToFoot	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.
TwoSidedHeadToHead	Page contents SHALL be imposed on the front and back sides so that the head (top) of the page contents back up to each other.

A.2.39 SourceColorSpace

SourceColorSpace specifies the color space that is to be operated on.

Table A.39: SourceColorSpace Enumeration Values (Sheet 1 of 2)

VALUE	DESCRIPTION
All	Operates on all source color spaces. This is useful when specifying a convert operation using all PDL source-supplied characterizations with an XJDF supplied final target device profile.
CalGray	Defines a calibrated device independent representation of gray.
Calibrated	Operates on "CalGray" and "CalRGB" color spaces.
CalRGB	Defines a calibrated device independent representation of RGB.
CIEBased	Operates on CIE based color spaces; CIEBasedA , CIEBasedABC , CIEBasedDEF and CIE-BasedDEFG .
CMYK	Operates on all CMYK color spaces. This includes both characterized and uncharacterized CMYK color spaces.
DeviceCMYK	Operates on uncharacterized CMYK color spaces.
DeviceGray	Operates on uncharacterized gray color spaces.

Table A.39: SourceColorSpace Enumeration Values (Sheet 2 of 2)

VALUE	DESCRIPTION
DeviceN	Identifies the source color encoding as a "DeviceN" color space. The specific "DeviceN" color space to operate on is defined in the ColorantControl/DeviceNSpace resource. If "DeviceN" is specified, then ColorantControl/DeviceNSpace SHALL also be present.
DeviceRGB	Operates on uncharacterized RGB color spaces.
Gray	Operates on all gray color spaces. This includes both characterized and uncharacterized gray color spaces.
ICCBased	Operates on color spaces defined using ICC profiles. The "ICCBased" value includes EPS, TIFF or PICT files with embedded ICC profiles. See ▶ [ICC.1]. It also includes PDF device color spaces that are characterized in ▶ table footnote #b of ▶ Table A.40 Mapping of SourceColorSpace enumerations to color spaces in the most common input file formats.
ICCCMYK	Operates on ICCBased color spaces with ICC CMYK profiles or DeviceCMYK having an ICC-based characterization. See ▶ table footnote #b of ▶ Table A.40 Mapping of SourceColorSpace enumerations to color spaces in the most common input file formats.
ICCGray	Operates on ICCBased color spaces with ICC gray profiles or DeviceGray having an ICC-based characterization. See ▶ table footnote #b of ▶ Table A.40 Mapping of SourceColorSpace enumerations to color spaces in the most common input file formats.
ICCLAB	Operates on an ICCBased device independent representation of Lab.
ICCRGB	Operates on ICCBased color spaces with ICC RGB profiles or DeviceRGB having an ICC-based characterization. See ▶ table footnote #b of ▶ Table A.40 Mapping of SourceColorSpace enumerations to color spaces in the most common input file formats.
Lab	Operates on Lab color spaces.
RGB	Operates on all RGB color spaces. This includes both characterized and uncharacterized RGB color spaces.
Separation	Operates on separation color spaces (spot colors). The specific separation(s) to operate on are defined in the @Separations attribute. If @Separations is not defined, the operation will operate on all the separation color spaces in the input RunList .
YUV	Operates on Yuv color spaces (also known as YCbCr). See ▶ [BT.601-7].

A.2.39.1 Source color space mapping

This table summarizes how the color spaces in ▶ Table A.2.39 SourceColorSpace above SHALL be mapped to/from different file formats.

Table A.40: Mapping of SourceColorSpace enumerations to color spaces in the most common input file formats (Sheet 1 of 3)

SOURCECS	FILE FORMAT	COLOR SPACE
Calibrated	PDF ^a	CalGray, CalRGB
	PostScript ^a	n/a
	TIFF	n/a
CIEBased	PDF ^a	n/a
	PostScript ^a	CIEBasedABC, CIEBasedA, CIEBasedDEF and CIEBasedDEFG
	TIFF	n/a

Table A.40: Mapping of SourceColorSpace enumerations to color spaces in the most common input file formats (Sheet 2 of 3)

SOURCECS	FILE FORMAT	COLOR SPACE
CMYK	PDF ^a	DeviceCMYK ^b PDF ICCBased color spaces with ICC CMYK profiles. CIEBasedDEFG spaces that resolve to a characterized CMYK space.
	PostScript ^a	DeviceCMYK
	TIFF	PhotometricInterp = 5 Samples per pixel = 4
DeviceCMYK	PDF ^a	DeviceCMYK ^b
	PostScript ^a	DeviceCMYK
	TIFF	PhotometricInterp = 5 Samples per pixel = 4
DeviceGray	PDF ^a	DeviceGray ^b
	PostScript ^a	DeviceGray
	TIFF	PhotometricInterp = 0 or 1
DeviceN	PDF ^a	DeviceN
	PostScript ^a	DeviceN
	TIFF	PhotometricInterp = 5, Samples per pixel = N
DeviceRGB	PDF ^a	DeviceRGB ^b
	PostScript	DeviceRGB
	TIFF	PhotometricInterp = 2
Gray	PDF ^a	DeviceGray ^b PDF ICCBased color spaces with ICC gray profiles. CIEBasedA spaces that resolve to a characterized gray space.
	PostScript ^a	DeviceGray
	TIFF	PhotometricInterp = 0 or 1
ICCBased ICCMYK ICCGray ICCLAB ICCRGB	PDF ^a	ICCBased DeviceGray ^b , DeviceCMYK ^b , DeviceRGB ^b
	PostScript ^a	n/a
	PostScript/EPS	The EPS file has an embedded ICC profile.
	TIFF	The TIFF file has an embedded ICC profile.
Lab	PDF ^a	Lab
	PostScript ^a	n/a
	TIFF	PhotometricInterp = 8 (CIELAB 1976 “normal” encoding) or PhotometricInterp = 9 (CIELAB 1976 using ICC profile v2 encoding).

Table A.40: Mapping of SourceColorSpace enumerations to color spaces in the most common input file formats (Sheet 3 of 3)

SOURCECS	FILE FORMAT	COLOR SPACE
RGB	PDF ^a	DeviceRGB ^b PDF ICCBased color spaces with ICC RGB profiles. CIEBasedDEF spaces that resolve to a characterized RGB space.
	PostScript	DeviceRGB
	TIFF	PhotometricInterp = 2
Separation	PDF ^a	Separation
	PostScript ^a	Separation
	TIFF	PhotometricInterp = 5 (Applies only to one of the planes in the separated image.)
YUV	PDF ^a	n/a
	PostScript ^a	n/a
	TIFF	PhotometricInterp = 6

- Where a **Pattern** or **Indexed** color space has been used in the PDL, the base color space is used to determine whether to apply this operation.
- In PDF, DeviceCMYK, DeviceRGB and DeviceGray source color spaces 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 such cases, the resulting color space is considered characterized by **XJDF** operations.

A.2.40 SourceObjects

SourceObjects specifies the class of a graphical object. Multiple tokens specify that the action that is filtered by

- ▶ SourceObjects applies to all of the listed classes.

Table A.41: SourceObjects Enumeration Values

VALUES	DESCRIPTION
ImagePhotographic	Contone images.
ImageScreenShot	Images largely comprised of rasterized vector art.
LineArt	Vector objects other than text.
SmoothShades	Gradients and blends.
Text	Text objects.

A.2.41 StapleShape

StapleShape specifies the required shape of the finished staple used for **Stitching**.

Table A.42: StapleShape Enumeration Values (Sheet 1 of 2)






VALUES	DESCRIPTION
Butted	
ClinchOut	

Table A.42: StapleShape Enumeration Values (Sheet 2 of 2)

VALUES	DESCRIPTION
Crown	
Eyelet	
Overlap	

A.2.42 Status

Status specifies the state of a process or queue entry that is required to execute a given task.

Table A.43: Status Enumeration Values

VALUE	DESCRIPTIONS
Aborted	Indicates that the process executing the XJDF has been aborted, which means that execution will not be resumed again.
Cleanup	The process represented by this node is currently being cleaned up.
Completed	Indicates that the node or queue entry has been executed correctly, and is finished.
InProgress	The node is currently executing.
Setup	The process represented by this node is currently being set up.
Stopped	Execution has been stopped. If a job is "Stopped", 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 to the job to be removed from the device.
Suspended	Execution has been stopped. If a job is "Suspended", running will be resumed later. Unlike "Stopped" this status indicates that the job is no longer blocking resources on the device, and other jobs may be run on the device. For instance, a job that has been ripped on a DFE and is waiting for the marker is "Suspended". When resumed, the job MAY go into @Status="Setup" before changing to "InProgress" again. The value "Suspended" is also used to describe iterations. In an iterative environment, "Suspended" specifies that at least one iteration cycle has completed but additional iteration cycles MAY still occur. In this case, @StatusDetails SHOULD be set to "IterationPaused".
Waiting	The node can be executed.

A.2.43 TightBacking

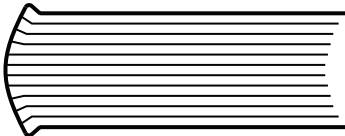
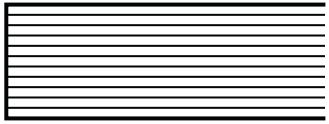
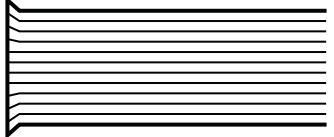
TightBacking specifies the required geometry for the back of a book block.

Note: This table is not ordered alphabetically - it is ordered by the pressure required, lowest first.

Table A.44: TightBacking Enumeration Values (Sheet 1 of 2)

VALUES	DESCRIPTION	BOOK FORM
Round	Rounding way.	

Table A.44: TightBacking Enumeration Values (Sheet 2 of 2)

VALUES	DESCRIPTION	BOOK FORM
RoundBacked	Rounding way, backing way.	
Flat	A flat backing - no tight backing is applied.	
FlatBacked	Backing way.	

A.2.44 UpdateGranularity

UpdateGranularity specifies which *QueueEntry* items in a *Queue* are to be included in any action.

Table A.45: UpdateGranularity Enumeration Values

VALUES	DESCRIPTION
All	All <i>QueueEntry</i> elements SHOULD be included.
ChangesOnly	Only those <i>QueueEntry</i> elements that have new information since the action SHOULD be included.

A.2.45 Usage

Usage specifies how a resource SHALL be used by a process.

Table A.46: Usage Enumeration Values

VALUES	DESCRIPTION
Input	The resource SHALL be used as an input.
Output	The resource SHALL be used as an output.

A.2.46 WorkingDirection

WorkingDirection specifies the direction of an action or of the application of a resource.

Table A.47: WorkingDirection Enumeration Values

VALUES	DESCRIPTION
Bottom	From below.
Top	From above.

A.2.47 WorkStyle

WorkStyle specifies the style of working in a sheet fed press. It is defined in the press coordinate system, where the sheet moves parallel to the Y axis. In the simple case of a single unrotated page per surface this implies that a flip around the

Y-axis (WorkAndTurn, WorkAndBack) will result in head to head images for the back side, whereas a flip around the X-axis (WorkAndTumble) will result in head to foot images.

Table A.48: WorkStyle Enumeration Values

VALUE	DESCRIPTION	TURN AXIS	PLATE REUSE
Perfecting	This work style describes the printing on both sides of the substrate using a separate set of plates for each side. The front lay is altered by flipping the sheet along the X-axis and thus retaining the side lays. Note: Perfecting is geometrically very similar to WorkAndTumble. Note: Perfecting is most commonly used with in-press perfecting units but the sheets can also be flipped outside of the press. The name Perfecting was chosen mainly for backwards compatibility.	X-Axis	No
Simplex	No turning.	None	No
WorkAndBack	This work style describes the printing on both sides of the substrate with different images or plate sets. After the first run the side lays are altered by flipping the sheet along the Y-axis and thus retaining the front lays. Note: WorkAndBack is geometrically very similar to WorkAndTurn.	Y-Axis	No
WorkAndTumble	This work style describes the printing on both sides of the substrate using the same set of plates for both surfaces. After the first run the front lay is altered by flipping the sheet along the X-axis and thus retaining the side lays. Note: WorkAndTumble SHOULD NOT be specified for digital printing.	X-Axis	Yes
WorkAndTurn	This work style describes the printing on both sides of the substrate with the same images or plate set on each side. After the first run the side lays are altered by flipping the sheet along the Y-axis and thus retaining the front lays. Note: WorkAndTurn SHOULD NOT be specified for digital printing.	Y-Axis	Yes

A.3 Preferred String and NMTOKEN Values

This section contains the preferred values for items of type string or NMTOKEN. Although these types are open lists, the values in these tables SHOULD be used where possible.

A.3.1 Comb and Coil Shapes

When specifying the shape of a comb or coil for LooseBinding, values from the following table are recommended.

Table A.49: Comb and Coil Shapes

VALUE	DESCRIPTION
Single	Each “tooth” is made with one wire.
SingleCalendar	Each “tooth” is made with one wire and an extension for hanging the bound product is provided in the center.
Twin	The shape of each “tooth” is made with a double wire (e.g., Wire-O®).
TwinCalendar	The shape of each “tooth” is made with a double wire and an extension for hanging the bound product is provided in the center.

A.3.2 Contact Types

When specifying the role of a contact, values from the following table are recommended.

Table A.50: Contact Types

VALUE	DESCRIPTION
Accounting	Contact information that relates to the invoice.

Table A.50: Contact Types

VALUE	DESCRIPTION
Administrator	Person to contact for queries concerning the execution of the job.
Agency	The contact is an employee of an agency.
Approver	The person who approves the job.
ArtDelivery	Delivery contact for artwork of the job.
ArtReturn	Return delivery contact for artwork of this job.
Author	
Customer	The end customer.
Delivery	The delivery address for all products of the job.
DeliveryCharge	The contact who is charged for delivery of the job.
Designer	
Editor	
Employee	Employee who works for the company processing the job.
Illustrator	
Owner	The owner of a resource.
Photographer	
Recipient	The contact is a recipient of a variable data record.
SenderAlias	The sender address that SHALL be printed on a delivery to an end customer. Note: This allows a company that has contracted out the work to hide the subcontractor.
TelephoneSanitizer	

A.3.3 Content Types

When specifying the type of content required or delivered, values from the following table are recommended.

Table A.51: Content Types

VALUE	DESCRIPTION
Ad	A single advertisement.
Article	A single article, including headers, text bodies, photos etc.
Barcode	A barcode.
Composed	A combination of elements that define an element that is not bound to a document page.
Editorial	An element that contains editorial matter, e.g. text, photographs etc.
Graphic	An element that contains line art.
IdentificationField	A general identification field excluding bar codes.
Image	A bitmap image.
Page	A representation of one document page.
Surface	A representation of an imposed surface.
Text	Formatted or unformatted text.

A.3.4 Delivery Methods

Delivery methods specify the recommended values for requesting how items are to be delivered.

Table A.52: Delivery Methods

VALUE	DESCRIPTION
Courier	
CourierNoSignature	A delivery service that does not require receipt stamps at the recipient's mailbox and/or mail room. This value is compatible with the commonly used Japanese 'Mail Bin' delivery service.
Email	
ExpressMail	
Ground	
InstantMessaging	
InterofficeMail	
Local	The items are already in place, no other delivery process is required.
NetworkCopy	This include LAN and VPN.
Storage	The item is stored by the supplier.

A.3.5 Device Classes

CIP4 supports many device classes. The following values SHOULD be used when filling `Device/@DeviceClass`.

Table A.53: Device Classes (Sheet 1 of 3)

VALUE	DESCRIPTION
CaseMaker	A case maker produces the hard case for books. See ▶ Section 5.6.5 CaseMaking.
Controller	A controller is a device that is a proxy for one or more individual devices or machines.
Cutter	A cutter can be used either to cut sheet blocks from a sheet fed press or to 'slit' a ribbon from a web fed press. See ▶ Section 5.6.10 Cutting.
DieCutter	A die cutter can be used to cut shapes from printed sheet blocks, e.g. windows in envelopes. See ▶ Section 5.6.27 ShapeCutting.
EndsheetFeeder	An end sheet feeder adds end sheets to a cover prior to binding. See ▶ Section 5.6.14 Feeding.
FilmSetter	A film setter creates a printable image on film. See ▶ Section 5.4.7 ImageSetting.
Folder	A folder can be used to fold the output from either sheet or web fed presses. See ▶ Section 5.6.15 Folding.
Gatherer	A gatherer can be used to collect sheets into piles. See ▶ Section 5.6.16 Gathering.
GathererBinder	A gatherer binder can be used to collect sheets into collated piles that are then bound. See ▶ Section 5.6.16 Gathering and ▶ Section 5.6.24 LooseBinding.
Hardcover	This device creates a hardcover. See ▶ Section 5.6.5 CaseMaking.

Table A.53: Device Classes (Sheet 2 of 3)

VALUE	DESCRIPTION
HardcoverBookLine	This device combines multiple processes to create and apply a hardcover to a block that it creates from a set of pages. See ▶ Section 5.6.1 BlockPreparation, ▶ Section 5.6.5 CaseMaking, ▶ Section 5.6.6 CasingIn, ▶ Section 5.6.7 Collecting, ▶ Section 5.6.8 CoverApplication, ▶ Section 5.6.13 EndSheetGluing, ▶ Section 5.6.16 Gathering, ▶ Section 5.6.17 Gluing, ▶ Section 5.6.18 HeadBandApplication, ▶ Section 5.6.21 Jacketing, ▶ Section 5.6.29 SpinePreparation and ▶ Section 5.6.30 SpineTaping.
HolePuncher	A hole puncher can be used to stamp or drill a number of holes, usually in a block of pages. See ▶ Section 5.6.19 HoleMaking.
Insertter	An inserter can be used to insert a component within another component. See ▶ Section 5.6.20 Inserting.
IntegratedDigitalPrinter	A digital printer that has additional post press capabilities, such as folding or binding. See ▶ Section 5.5.2 DigitalPrinting.
Jacketer	A jacketer wraps a bound book with a folded jacket. See ▶ Section 5.6.21 Jacketing.
MultipleWebConventionalPress	A multiple web conventional press. See ▶ Section 5.5.1 ConventionalPrinting.
PerfectBinder	This device combines multiple processes to create perfect bound books. See ▶ Section 5.6.1 BlockPreparation, ▶ Section 5.6.5 CaseMaking, ▶ Section 5.6.6 CasingIn, ▶ Section 5.6.7 Collecting, ▶ Section 5.6.8 CoverApplication, ▶ Section 5.6.13 EndSheetGluing, ▶ Section 5.6.16 Gathering, ▶ Section 5.6.17 Gluing, ▶ Section 5.6.18 HeadBandApplication, ▶ Section 5.6.21 Jacketing, ▶ Section 5.6.29 SpinePreparation and ▶ Section 5.6.30 SpineTaping.
PerfectBinderLine	This device combines multiple processes to create perfect bound books. See ▶ Section 5.6.1 BlockPreparation, ▶ Section 5.6.5 CaseMaking, ▶ Section 5.6.6 CasingIn, ▶ Section 5.6.7 Collecting, ▶ Section 5.6.8 CoverApplication, ▶ Section 5.6.13 EndSheetGluing, ▶ Section 5.6.16 Gathering, ▶ Section 5.6.17 Gluing, ▶ Section 5.6.18 HeadBandApplication, ▶ Section 5.6.21 Jacketing, ▶ Section 5.6.29 SpinePreparation and ▶ Section 5.6.30 SpineTaping.
PlateSetter	A plate setter creates a printable image on a plate suitable for conventional printing. See ▶ Section 5.4.7 ImageSetting.
PrintingPress	Any type of printing press. See ▶ Section 5.5.1 ConventionalPrinting and ▶ Section 5.5.2 DigitalPrinting.
Scanner	A scanner is used to describe the manual process of producing machine readable image data from pre-printed documents. See ▶ Section 5.3.3 ManualLabor.
SheetFedConventionalPress	A standard sheet fed conventional press. See ▶ Section 5.5.1 ConventionalPrinting.
SheetFedDigitalPrinter	A standard sheet fed digital press. See ▶ Section 5.5.2 DigitalPrinting.
SingleWebConventionalPress	A single web conventional press. See ▶ Section 5.5.1 ConventionalPrinting.
Stacker	A stacker can be used to create a pile or bundle of components suitable for delivery. See ▶ Section 5.6.31 Stacking.

Table A.53: Device Classes (Sheet 3 of 3)

VALUE	DESCRIPTION
Stitcher	A stitcher can be used to stitch a number of sheets together into a block and may also add a cover. See ▶ Section 5.6.32 Stitching.
ThreadSewer	A thread sewer can be used to sew a number of sheets together into a block. See ▶ Section 5.6.35 ThreadSewing.
Trimmer	A trimmer can be used to reduce a block to the required size, e.g. for subsequent hard-cover binding. See ▶ Section 5.6.36 Trimming.
WebDigitalprinter	A single web digital press. See ▶ Section 5.5.2 DigitalPrinting.
WideFormatPrinter	A wide format printer can be used to create large printed products such as banners. See ▶ Section 5.5.2 DigitalPrinting.

A.3.6 Flute Types

Values of this type define the required flute type (size and frequency) for 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. See ▶ [Corrugated Packaging].

Table A.54: Flute Types

VALUE	DESCRIPTION
A	33±3 flutes/foot, 108±10 flutes/meter.
B	47±3 flutes/foot, 154±10 flutes/meter.
C	39±3 flutes/foot, 128±10 flutes/meter.
E	90±4 flutes/foot, 295±13 flutes/meter.
F	125±4 flutes/foot, 420±13 flutes/meter.

A.3.7 Fold Catalog

Fold catalog describes a type of fold according to the folding catalog in ▶ Figure A-1: Fold catalog . In case of any ambiguity, the folding notation SHALL take precedence over the graphic illustration in the aforementioned figure.

The value format is: "Fn-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., "F6-2" describes a Z-fold of 6 finished pages, and "F6-X" describes a generic fold with 6 finished pages).

Figure A-1: Fold catalog

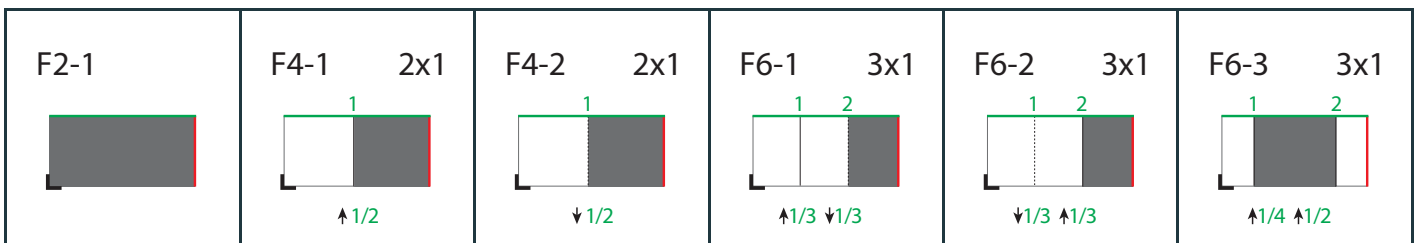


Figure A-1: Fold catalog

<p>F6-4 3x1</p> <p>↑1/3 ↑1/3</p>	<p>F6-5 3x1</p> <p>↑2/3 ↓1/3</p>	<p>F6-6 3x1</p> <p>↑3/4 ↓1/4</p>	<p>F6-7 3x1</p> <p>↑1/4 ↓1/4</p>	<p>F6-8 3x1</p> <p>↑2/3 ↑1/3</p>	<p>F8-1 4x1</p> <p>↑1/2 ↑1/4</p>
<p>F8-2 4x1</p> <p>↑1/2 ↓1/4</p>	<p>F8-3 4x1</p> <p>↑1/4 ↓1/4 ↑1/4</p>	<p>F8-4 4x1</p> <p>↑1/4 ↑1/2 ↓1/4</p>	<p>F8-5 4x1</p> <p>↑1/4 ↑1/4 ↑1/4</p>	<p>F8-6 4x1</p> <p>↑3/4 ↓1/4 ↓1/4</p>	<p>F8-7 2x2</p> <p>↑1/2 + ↑1/2</p>
<p>F10-1 5x1</p> <p>↑1/5 ↓1/5 ↑1/5 ↓1/5</p>	<p>F10-2 5x1</p> <p>↑4/5 ↓1/5 ↓1/5 ↓1/5</p>	<p>F10-3 5x1</p> <p>↑2/5 ↓2/5 ↑1/5</p>	<p>F12-1 6x1</p> <p>↑1/3 ↓1/3 ↑1/6</p>	<p>F12-2 6x1</p> <p>↑1/3 ↑1/3 ↓1/6</p>	<p>F12-3 6x1</p> <p>↑1/2 ↓1/6 ↑1/6</p>
<p>F12-4 6x1</p> <p>↑1/2 ↓1/6 ↓1/6</p>	<p>F12-5 6x1</p> <p>↑1/2 ↓1/3 ↑1/6</p>	<p>F12-6 6x1</p> <p>↑1/6 ↓1/6 ↑1/6 ↓1/6 ↑1/6</p>	<p>F12-7 3x2</p> <p>↑1/3 ↓1/3 + ↑1/2</p>	<p>F12-8 3x2</p> <p>↑2/3 ↑1/3 + ↑1/2</p>	<p>F12-9 3x2</p> <p>↑1/3 ↑1/3 + ↑1/2</p>
<p>F12-10 3x2</p> <p>↑2/3 ↓1/3 + ↑1/2</p>	<p>F12-11 3x2</p> <p>↑1/3 + ↑1/2 + ↑1/3</p>	<p>F12-12 2x3</p> <p>↑1/2 + ↑2/3 ↓1/3</p>	<p>F12-13 2x3</p> <p>↑1/2 + ↑1/3 ↑1/3</p>	<p>F12-14 2x3</p> <p>↑1/2 + ↑1/3 ↓1/3</p>	<p>F14-1 7x1</p> <p>↑1/7 ↓1/7 ↑1/7 ↓1/7 ↑1/7 ↓1/7</p>
<p>F16-1 8x1</p> <p>↑1/2 ↓1/4 ↑1/8</p>	<p>F16-2 8x1</p> <p>↑1/2 ↓1/4 ↓1/8</p>	<p>F16-3 8x1</p> <p>↑1/2 ↑1/4 ↓1/8</p>	<p>F16-4 8x1</p> <p>↑1/2 ↑1/4 ↑1/8</p>	<p>F16-5 8x1</p> <p>↓1/8 ↑1/8 ↓1/8 ↑1/8 ↓1/8 ↑1/8 ↓1/8</p>	<p>F16-6 4x2</p> <p>↑1/2 + ↑1/2 + ↑1/4</p>
<p>F16-7 4x2</p> <p>↑1/2 + ↑1/2 + ↓1/4</p>	<p>F16-8 4x2</p> <p>↑1/2 + ↓1/2 + ↓1/4</p>	<p>F16-9 4x2</p> <p>↑1/2 ↓1/4 + ↑1/2</p>	<p>F16-10 4x2</p> <p>↑1/2 ↑1/4 + ↑1/2</p>	<p>F16-11 4x2</p> <p>↑1/4 ↓1/4 ↑1/4 + ↑1/2</p>	<p>F16-12 4x2</p> <p>↑1/4 ↑1/4 ↑1/4 + ↑1/2</p>
<p>F16-13 2x4</p> <p>↑1/2 + ↑1/2 ↓1/4</p>	<p>F16-14 2x4</p> <p>↑1/2 + ↑1/2 ↑1/4</p>	<p>F18-1 9x1</p> <p>↑1/9 ↓1/9 ↑1/9 ↓1/9 ↑1/9 ↓1/9 ↑1/9 ↓1/9</p>	<p>F18-2 9x1</p> <p>↑2/3 ↓1/3 ↑1/9 ↓1/9</p>	<p>F18-3 9x1</p> <p>↑1/3 ↓1/3 ↑2/9 ↓1/9</p>	<p>F18-4 9x1</p> <p>↑1/3 ↓1/3 ↑1/9 ↓1/9</p>

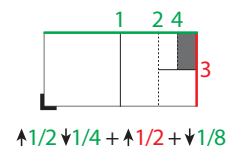
Figure A-1: Fold catalog

<p>F18-5 3x3</p> <p>↑1/3 ↓1/3 + ↑1/3 ↓1/3</p>	<p>F18-6 3x3</p> <p>↑1/3 ↓1/3 + ↑2/3 ↓1/3</p>	<p>F18-7 3x3</p> <p>↑1/3 ↑1/3 + ↑1/3 ↓1/3</p>	<p>F18-8 3x3</p> <p>↑1/3 ↑1/3 + ↑2/3 ↓1/3</p>	<p>F18-9 3x3</p> <p>↑2/3 ↑1/3 + ↑2/3 ↑1/3</p>	<p>F20-1 5x2</p> <p>↑2/5 ↓2/5 ↑1/5 + ↑1/2</p>
<p>F20-2 5x2</p> <p>↑1/5 ↓1/5 ↑1/5 ↓1/5 + ↑1/2</p>	<p>F24-1 6x2</p> <p>↑1/3 ↓1/3 + ↑1/2 + ↑1/6</p>	<p>F24-2 6x2</p> <p>↑1/3 ↑1/3 + ↑1/2 + ↑1/6</p>	<p>F24-3 6x2</p> <p>↑1/3 ↓1/3 ↑1/6 + ↑1/2</p>	<p>F24-4 6x2</p> <p>↑1/3 ↓1/3 ↓1/6 + ↑1/2</p>	<p>F24-5 6x2</p> <p>↑1/3 ↑1/3 ↓1/6 + ↑1/2</p>
<p>F24-6 6x2</p> <p>↑1/6 ↓1/6 ↑1/6 ↓1/6 + ↑1/2</p>	<p>F24-7 6x2</p> <p>↑1/3 + ↑1/2 + ↑1/3 ↓1/6</p>	<p>F24-8 3x4</p> <p>↑1/3 ↓1/3 + ↑1/2 ↓1/4</p>	<p>F24-9 3x4</p> <p>↑2/3 ↑1/3 + ↑1/2 ↓1/4</p>	<p>F24-10 3x4</p> <p>↑1/3 ↑1/3 + ↑1/2 ↓1/4</p>	<p>F24-11 4x3</p> <p>↑1/2 + ↑2/3 ↓1/3 + ↑1/4</p>
<p>F28-1 7x2</p> <p>↑1/7 ↓1/7 ↑1/7 ↓1/7 + ↑1/2</p>	<p>F32-1 16x1</p> <p>↑1/2 ↓1/4 ↑1/8 ↓1/16</p>	<p>F32-2 8x2</p> <p>↑1/2 ↓1/4 + ↑1/2 ↑1/8</p>	<p>F32-3 8x2</p> <p>↑1/2 ↓1/4 + ↑1/2 + ↓1/8</p>	<p>F32-4 4x4</p> <p>↑1/2 + ↑1/2 + ↑1/4 + ↑1/4</p>	<p>F32-5 4x4</p> <p>↑1/2 + ↑1/2 + ↓1/4 + ↓1/4</p>
<p>F32-6 4x4</p> <p>↑1/2 + ↑1/2 + ↑1/4 + ↓1/4</p>	<p>F32-7 4x4</p> <p>↑1/4 ↓1/4 ↑1/4 + ↑1/2 ↓1/4</p>	<p>F32-8 4x4</p> <p>↑1/2 ↓1/4 + ↑1/2 ↓1/4</p>	<p>F32-9 4x4</p> <p>↑1/2 + ↑1/2 ↓1/4 + ↑1/4</p>	<p>F36-1 9x2</p> <p>↑1/3 ↓1/3 ↑1/9 ↓1/9 + ↑1/2</p>	<p>F36-2 6x3</p> <p>↑1/3 ↓1/3 + ↑1/3 ↓1/3 + ↑1/6</p>
<p>F40-1 5x4</p> <p>↑1/5 ↓1/5 ↑1/5 ↓1/5 + ↑1/2 ↓1/4</p>	<p>F48-1 6x4</p> <p>↑1/3 ↓1/3 + ↑1/4 ↓1/4 + ↑1/6</p>	<p>F48-2 4x6</p> <p>↑1/4 ↓1/4 ↑1/4 + ↑1/3 ↓1/3 + ↑1/6</p>	<p>F64-1 8x4</p> <p>↑1/2 + ↑1/4 ↓1/4 + ↑1/4 + ↑1/4 ↓1/8</p>	<p>F64-2 8x4</p> <p>↑1/4 ↓1/4 ↑1/4 + ↑1/4 ↓1/4 + ↑1/4 + ↑1/8</p>	

Legend

- Fold up
- Fold down
- Fold direction change, 90° (alternating)
- Finished format folded sheet
- 1, 2, 3 Folds in numeric order
- Lay
- Green: open sheet length
- Red: open sheet width

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° clockwise
- ↑1/2 Fold up with 1/2 of the open sheet format
- + : Fold direction change: 90° counterclockwise
- ↓1/8 Fold down with 1/8 of the open sheet format length

A.3.8 Ink and Varnish Coatings

When specifying coating types, such as ink or varnish, values from the following table are recommended.

Table A.55: Ink and Varnish Coatings

VALUE	DESCRIPTION
Aqueous	Water based coating.
Gloss	A glossy coating.
Ink	Any generic ink.
InkJet	Ink.
Latex	Liquid that is similar to ink.
Matte	A matte coating.
Primer	A coating that is applied beneath the image.
Relief	Property of the coating.
RubResistant	Attribute of the ink.
Satin	A coating between Gloss and Matte .
Silicone	Liquid that is similar to ink.
Toner	Liquid that is similar to ink.
UV	Ultra violet cured polymers.
Varnish	Unpigmented ink.
WaterResistant	Attribute of the ink.

A.3.9 Input Tray and Output Bin Names

Part/@Location MAY be used to specify a location within a device (e.g., a paper tray). When specifying input paper trays (indicated with “I”) and/or output bins (indicated with “O”), the following values for **Part/@Location** SHOULD be used.

Table A.56: Input Tray and Output Bin Names (Sheet 1 of 2)

VALUE	I/O	DESCRIPTION
AnyLargeFormat	IO	The location that holds larger format media with one dimension larger than 11 inches. The media dimensions SHALL be specified.
AnySmallFormat	IO	The location that holds smaller format media. The media dimensions SHALL be specified.
AutoSelect	IO	The location that the device selects based on the Media specification.
Booklet	O	The bin where the device places booklets.
Bottom	IO	The location that, when facing the device, can best be identified as ‘bottom’.
BypassTray	I	The input tray used to handle odd or special papers. MAY be used to specify the input tray that is used for insert sheets that are not to be imaged.
BypassTray-N	I	The input tray used to handle odd or special papers. MAY be used to specify the input tray that is used for insert sheets that are not to be imaged. N = ‘1’, ‘2’, ...
Continuous	IO	The location to handle continuous media (i.e., continuously connected sheets).
Disc	IO	The location to handle CD or DVD discs to be printed on.
Disc-N	IO	The location to handle CD or DVD discs to be printed on. N = ‘1’, ‘2’, ...
Envelope	IO	The location to handle envelopes.

Table A.56: Input Tray and Output Bin Names (Sheet 2 of 2)

VALUE	I/O	DESCRIPTION
Envelope-N	IO	The location to handle envelopes. N = '1', '2', ...
Front	IO	The location that, when facing the device, can best be identified as 'front.'
InsertTray	I	The input tray that can best be identified as 'insert tray.' Used to specify the input tray that is used for insert sheets (insert sheets are never imaged).
InsertTray-N	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 insert sheets (insert sheets are never imaged).
LargeCapacity	IO	The location that can best be identified as the 'large capacity' location (in terms of the number of sheets) with respect to the device.
LargeCapacity-N	IO	The location that can best be identified as the 'large capacity-1', 'large-capacity-2', ... etc., input or output location (in terms of the number of sheets) with respect to the device.
Left	IO	The bin that, when facing the device, can best be identified as 'left.'
Mailbox-N	O	The output location that is best identified as "Mailbox #1", "Mailbox #2", etc.
Middle	IO	The location that, when facing the device, can best be identified as "middle".
PostMarkerInserter	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.
Rear	IO	The location that, when facing the device, can best be identified as "rear".
Right	IO	The location that, when facing the device, can best be identified as "right".
Roll	IO	The location to handle web-fed media.
Roll-N	IO	The Nth location to handle the Nth web-fed media.
Side	IO	The location that, when facing the device, can best be identified as "side".
Stacker-N	O	The output location that is best identified as "Stacker #1", "Stacker #2", etc.
Top	IO	The location that, when facing the device, can best be identified as "top".

A.3.10 MediaType Details

MediaType Details specifies additional details of the media to be used.

Table A.57: MediaType Details (Sheet 1 of 2)

VALUE	DESCRIPTION
Aluminum	Conventional or CTP press plate.
Backlit	Any media that is designed to be illuminated from the back side.
Cardboard	
CD	CD disc to be printed on.
Cloth	Cloth, e.g. for a hardcover book case.
Continuous	Continuously connected sheets of an opaque material. The edge that is connected is not specified.
ContinuousLong	Continuously connected sheets of an opaque material connected along the long edge.
ContinuousShort	Continuously connected sheets of an opaque material connected along the short edge.
DoubleWall	Double-wall corrugated board.
DryFilm	

Table A.57: MediaType Details (Sheet 2 of 2)

VALUE	DESCRIPTION
DVD	DVD disc to be printed on.
EndBoard	End board used in the Bundling process.
Envelope	Envelopes that can be used for conventional mailing purposes.
EnvelopePlain	Envelopes that are not preprinted and have no windows.
EnvelopeWindow	Envelopes that have windows for addressing purposes.
EnvelopeWindowLeft	Envelopes that have windows on the left for addressing purposes.
EnvelopeWindowRight	Envelopes that have windows on the right for addressing purposes.
FlexoBase	For the base layer of flexo plates.
FlexoPhotoPolymer	For the photopolymer layer of flexo plates.
Flute	Flute layer of a corrugated board.
FullCutTabs	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.
ImageSetterPaper	Contact paper as replacement for film.
Labels	Label stock, e.g. a sheet of peel-off labels.
Leather	Leather stock, e.g. for a hardcover book case.
Letterhead	Separately cut sheets of an opaque material including a letterhead.
MultiLayer	Form medium composed of multiple layers that are attached to one another (e.g., for use with impact printers).
MultiPartForm	Form medium composed of multiple layers not attached to one another; each sheet might be drawn separately from an input source.
Photographic	Separately cut sheets of an opaque material to produce photographic quality images.
Polyester	Conventional or CTP press plate.
PreCutTabs	Media with tabs that are cut so that more than one tab is visible extending out beyond the edge of non-tabbed media.
ScrimBanner	Specific type of vinyl.
SingleFace	Single-face corrugated board.
SingleWall	Single-wall corrugated board.
Stationery	Separately cut sheets of an opaque material, includes generic paper.
TabStock	Media with tabs, either precut or full-cut.
Tractor	Tractor feed with holes.
Transparency	Separately cut sheets of a transparent material.
TripleWall	Triple-wall corrugated board.
WallPaper	Details of wall paper.
WetFilm	Conventional photographic film.

A.3.11 Milestones

The following table defines a list of values that are valid for [QueryNotification/NotificationFilter/@MilestoneTypes](#) and [Milestone/@MilestoneType](#).

Note: 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.

Table A.58: Milestones

VALUE	DESCRIPTION
BindingCompleted	All binding worksteps including packing of the job have been completed. Postpress worksteps are defined according to ▶ Section 5.6 Postpress Processes.
BindingInProgress	At least one of the binding worksteps of the job is in progress.
Delivered	The files were delivered to the destination.
DigitalArtArrived	Digital content has been received.
JobCompletedSuccessfully	Job completed successfully.
JobCompletedWithErrors	Job completed with errors.
JobCompletedWithWarnings	Job completed with warnings.
PageApproved	Planned page proofs have been approved.
PageCompleted	Pages are ready (no further page processing or page proofing required).
PageProofed	Planned page proofs have been made.
PostPressCompleted	All postpress worksteps including packing of the job have been completed. Postpress worksteps are defined according to ▶ Section 5.6 Postpress Processes.
PostPressInProgress	At least one of the postpress worksteps of the job is in progress.
PrePressCompleted	All prepress worksteps of the job have been completed. Prepress worksteps are defined according to ▶ Section 5.4 Prepress Processes. In conventional prepress, this is the case when all plates have been made.
PrePressInProgress	At least one of the prepress worksteps of the job is in progress.
PressCompleted	All press worksteps of the job have been completed. Press worksteps are defined according to ▶ Section 5.5 Press Processes.
PressInProgress	At least one of the press worksteps of the job is in progress.
ProofSent	Planned proofs sent to customer.
ShippingCompleted	Final product was delivered to the customer or distributors.
ShippingInProgress	Final product is being shipped.
SurfaceApproved	Planned imposition proofs have been approved.
SurfaceAssigned	Surfaces have their corresponding pages assigned (e.g., could be proofed).
SurfaceCompleted	Planned surfaces are ready (i.e., plates could be made).
SurfaceProofed	Planned imposition proofs have been made.

A.3.12 Module Types

The following tables provide lists of individual *Module/@ModuleType* values.

Table A.59: Module Types for Conventional Printing (Sheet 1 of 2)

VALUE	DESCRIPTION
CoatingModule	Unit for coatings, for example, full coating of varnish.
Delivery	Delivery module, unit for gathering the printed sheets.
Drier	Module for drying the previously printed color or varnish.

Table A.59: Module Types for Conventional Printing (Sheet 2 of 2)

VALUE	DESCRIPTION
ExtensionModule	Unit for extending the distance between modules, for example to increase the distance between the last printing module and the delivery module.
Feeder	Feeder module, feeds the device with paper.
Imaging	Imaging module in a direct to plate machine.
Numbering	Numbering unit.
PerfectingModule	Unit for perfecting, reversing device.
PrintModule	Unit for printing a color. Describes one cylinder and one side.

Table A.60: Module Types for Postpress (Sheet 1 of 2)

VALUE	DESCRIPTION
BlockPreparer	The block preparer prepares the book block for a hardcover book. See ▶ Section 5.6.1 BlockPreparation.
BoxFolder	The box folder folds and glues blanks into folded boxes for packaging. See ▶ Section 5.6.2 BoxFolding.
CaseMaker	The case maker produces the hard case for books. See ▶ Section 5.6.5 CaseMaking.
Caser	The caser joins the hardcover book case and the book block. See ▶ Section 5.6.6 CasingIn.
Chain	The transport chain or conveyer to transport gathered / collected product.
EndSheetGluer	The end sheet gluer merges the front end sheet, the book block and the back end sheet together. See ▶ Section 5.6.13 EndSheetGluing.
Feeder	The feeder module, feeds the device with paper. See ▶ Section 5.6.14 Feeding.
Gluer	The gluer applies glue to a component. See ▶ Section 5.6.17 Gluing.
HeadBandApplicator	The head band applicator applies a head band to the book block. See ▶ Section 5.6.18 HeadBandApplication.
InkJetPrinter	The printer that uses inkjet technology to print images or text on a component. See ▶ Section 5.5.2 DigitalPrinting.
Inserter	The inserter inserts one or more “child” components to one “mother” component. See ▶ Section 5.6.20 Inserting.
Jacketer	The jacketer wraps a jacket around a book. See ▶ Section 5.6.21 Jacketing.
PaperPath	The paper path module, path that paper follows through the machine.
PressingStation	The pressing station presses the cover to the book block.
ShapeCutter	The shape cutter produces special shapes like an envelope window or a heart-shaped beer mat. Note: The shape cutter module may contain tools that correspond to the actual dies. See ▶ Section 5.6.27 ShapeCutting.

Table A.60: Module Types for Postpress (Sheet 2 of 2)

VALUE	DESCRIPTION
SpinePreparer	The spine preparer prepares the spine of a book for hard and softcover production. See ▶ Section 5.6.29 SpinePreparation.
SpineTaper	The spine taper applies a tape strip to the spine of a book block. See ▶ Section 5.6.30 SpineTaping.
Strapper	The strapper straps a bundle of products. See ▶ Section 5.6.33 Strapping.
ThreadSealer	The thread sealer sews and seals a signature at the spine. See ▶ Section 5.6.34 ThreadSealing.
ThreadSewer	The thread sewer sews all signatures of a book block together. See ▶ Section 5.6.35 ThreadSewing.

Table A.61: Module Types for Digital Printing

VALUE	DESCRIPTION
FarmPrinter	Individual printer in a printer farm of printers.
Fuser	Fuser module — fuses the toner onto the media.
Marker	Marker module, excluding in-line finishing.
ReferencedDataCollector	Module that fetches data referenced from the XJDF and MAY include data referenced from the PDL. Does not include accepting a zip, unpacking a zip, or fetching the XJDF itself.
RIP	Raster image processor module.
Unpacker	Module that receives and unpacks the zip package and fetches the XJDF if it is referenced from the XJMF .

Table A.62: Module Types for Web Printing

VALUE	DESCRIPTION
ChillUnit	Chill unit that chills the heated printed paper.
ImprintUnit	Printing unit that allows changing plates during a production run, doing imprints.
PrintUnit	A print unit consists of multiple print module units.
RemoisteningModule	Module that can be used for high gloss varnish, re-moistened glue, rub-off ink or encapsulated fragrances. The re-moistening module is located between the last printing unit and the dryer.
UVCoater	The UV-Coater module applies UV-varnish with subsequent drying in a UV-dryer.

Table A.63: Module Types for Web Finishing (Sheet 1 of 2)

VALUE	DESCRIPTION
CrossCutter	Cuts the web/ribbon n-times into sheets and transports the sheets to inline postpress equipment.
Delivery	Delivers the printed and/or folded sheets out of the folder.
Folder	Module for cutting the collected ribbons into sheets, in some cases collecting these sheets, and folding the sheets (quarter and cross folds).

Table A.63: Module Types for Web Finishing (Sheet 2 of 2)

VALUE	DESCRIPTION
Former	Module for gathering ribbons and in most instances doing the first fold of the ribbons (quarter fold).
GluingAndSofteningModule	Consists of multiple heads, spread out in the press for gluing and/or softening of ribbons or folded sheets.
MoebiusDeinfinitizer	Used to resolve the infinite loops caused by printing on interleaving surfaces of Möbius banded webs.
PerforatingModule	Module for doing cross, longitudinal or diagonal perforations and die cuts on a web. The module is placed between chill unit and folder.
PlanoModule	The plano module cuts the web/ribbon into sheets and stacks the sheets into a pile.
PloughFoldModule	The plough fold module does a quarter fold to ribbons or webs and is mostly found in front of a folder module.
Rewinder	Rewinds the printed web to a roll.
RibbonCompensator	Controls the web's or ribbon's running direction regarding the cross cut.
Slitter	Module for cutting in the machine direction.
Stitcher	Stitches folded sheets together.
Superstructure	Module in which a web will be cut into ribbons that will then be moved to the correct position for folding.
TurnerBar	Turns the front side of a web to the back side and vice versa.
TurnerBarUnit	Turns the front side of a web to the back side and vice versa in a separate unit.

Table A.64: Module Types for Packing

VALUE	DESCRIPTION
BundlingModule	The bundling module is used for bundling components. See ▶ Section 5.6.4 Bundling.
LabelingModule	The labeling module is used for labeling a bundle. See ▶ Section 5.6.22 Labeling.
PalletizingModule	The palletizing module collects the bundles on a pallet. See ▶ Section 5.6.25 Palletizing.
Stacker	The stacker module stacks the component into a pile. See ▶ Section 5.6.31 Stacking.
Trimmer	The trimmer module trims the component to its final size. See ▶ Section 5.6.36 Trimming.

A.3.13 Node Categories

Node categories are used to indicate the general purpose of an **XJDF**.

Table A.65: Node Categories

VALUE	DESCRIPTION
Binding	Binding of a bound product.
Cutting	Specifies cutting of a Component .
DigitalPrinting	A RIP and print run on a digital printer that produces final output.
FinalImaging	A RIP and image that produces final output that is ready for further processing (e.g., film or plates).
FinalRIPing	A RIP process for generating final output.

Table A.65: Node Categories

VALUE	DESCRIPTION
Folding	Folding of a product.
Newsprinting	A press run on a news printing web press.
PostPress	General postpress. Includes "Folding" and "Binding".
PrePress	General prepress.
Printing	A press run that produces final output.
ProofImaging	A RIP that produces proof output.
ProofRIPing	A RIP process for generating a proof. The processes are identical to those specified for "FinalRIPing".
RIPing	General RIP gray box.
WebPrinting	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.
WebToPrint	A product description that describes a product order in a web shop.

A.3.14 Pallet Types

The following table defines a list of values that are valid for indicating the intended type of pallet to be used.

Table A.66: Pallet Types

VALUE	DESCRIPTION
2Way	Two-way entry.
4Way	Four-way entry.
Euro800x600	800mm x 600mm. See ▶ [DIN 15146-4] - equals half Euro pallet.
Euro800x1200	800mm x 1200mm. See ▶ [DIN EN 13698-1] - equals Euro pallet.
Euro1000x1200	1000mm x 1200mm. See ▶ [DIN EN 13698-2] - flat pallet.
Euro1200x1200	1200mm x 1200mm, no norm, but used in the field.

A.3.15 Printing Technologies

The following table defines a list of values that are valid for indicating the intended printing technology to be used.

Table A.67: Printing Technologies (Sheet 1 of 2)

VALUE	DESCRIPTION
DyeSublimation	For digital printing.
ElectroInk	Digital printing with liquid toner.
Electrophotography	Electrophotographic printing with toner.
Flexography	For conventional printing.
InkJet	For digital printing.
Latex	Specific type of inkjet.
Letterpress	Conventional printing with traditional relief masters.
OffsetLithography	For conventional printing.
Potato	Unconventional printing using a carved potato as a print master.
Rotogravure	For conventional printing.

Table A.67: Printing Technologies (Sheet 2 of 2)

VALUE	DESCRIPTION
ScreenPrinting	For conventional printing.
Thermal	For digital printing.
UV	For digital printing.

A.3.16 PrintStandard Characterization Data Sets

PrintStandard specifies the reference name of a characterization data set. There are research and trade associations (such as Fogra, IDEAlliance, WAN-IFRA, JPMA, ICC) that provide characterization data sets for standard printing conditions. Most reference names of standard printing conditions are registered with the ICC; see ▶ [Characterization Data].

Official reference names SHALL be taken if a standard printing condition exists. Custom or device dependent reference names MAY be provided if no official standard printing condition is available.

Note: In digital printing, PrintStandard will typically be used to specify the selected internal color model that defines the device specific use of colorants such as light cyan or additional gamut colors.

Note: Whereas PrintStandard defines a media independent characterization data set, *Part/@PrintCondition* defines a characterization data set that is applied to a specific setup including paper selection and screening setup.

Table A.68: PrintStandard Values

PRINTSTANDARD NAME	PROVIDER	DESCRIPTION
CGATS21-2-CRPC5	International Color Consortium	Valid for CGATS21-2-CRPC5 based profiles such as “SWOP2013C3-CRPC5”. See ▶ [CGATS.21].
CGATS21-2-CRPC6	International Color Consortium	Valid for CGATS21-2-CRPC6 based profiles such as “SWOP2013C3-CRPC5”. See ▶ [CGATS.21].
FOGRA39	FOGRA	Valid for FOGRA39L based profiles such as “ISO Coated v2 (ECI)” or “ISO Coated v2 300% (ECI)”. See ▶ [FOGRA].
FOGRA47	FOGRA	Valid for FOGRA47L based profiles such as “PSO Uncoated ISO 12647 (ECI)”. See ▶ [FOGRA].
FOGRA51	FOGRA	Valid for FOGRA51 based profiles such as “PSO Coated v3”. See ▶ [FOGRA].
FOGRA52	FOGRA	Valid for FOGRA52 based profiles such as “PSO Uncoated v3”. See ▶ [FOGRA].
FOGRA53	FOGRA	Valid for FOGRA53 based profiles such as “eciCMYK”. See ▶ [FOGRA].

A.3.17 Product Types

Table A.69: Product Types (Sheet 1 of 3)

VALUE	DESCRIPTION
BackCover	The last page or sheet of a softcover book or magazine, commonly a heavier media.
BlankBox	Cut, Unfolded box, input for folder-gluer
BlankSheet	An unprinted divider page or sheet. Also describes die-cut unprinted label.
BlankWeb	A web with connected blanks after a die cutting.
Body	Generic content inside of a cover, e.g. "BookBlock". Also, in page assembly, the main text content (body copy), in contrast to headings or front matter.
Book	Body with a cover and a spine.
BookBlock	The assembled body of pages for a hardcover book.

Table A.69: Product Types (Sheet 2 of 3)

VALUE	DESCRIPTION
BookCase	The assembled covers and spine component of a hardcover book, prior to "casing in" (attaching to the book block).
Booklet	Body with a cover without a spine (typically stapled).
Box	Convenience packaging that is not envisioned to be protection for shipping.
Brochure	A single folded sheet.
BusinessCard	A small card that displays contact information for an individual employed by a company.
Carton	Protection packaging for shipping.
Cover	A single sheet covering a side of a print product.
CoverLetter	A letter accompanying another print product.
EndSheet	A glued sheet that spans and attaches BookBlock to BookCase, in both front and back of a hardcover book, (printed or not).
Envelope	A folded paper container, with sealable flap, that encloses and protects a document or contents.
FlatBox	A folded and glued blank (not opened). Output from a box folder-gluer.
FlatWork	Non-bound, non-folded products or products that only have packaging folds.
FrontCover	The first page or sheet of a softcover book or magazine, commonly a heavier media.
Insert	A product part intended to be inserted into a print product.
Jacket	Hardcover case jacket.
Label	A piece of paper or plastic that is attached to an object in order to give information about it.
Leaflet	A single unfolded sheet.
Letter	A written or printed communication addressed to a person or organization and usually transmitted by mail or messenger.
Map	A drawing/representation of a particular area such as a city, or a continent, showing its main features, as they would appear if viewed from above.
Media	Unprinted media, the substrate (usually paper) on which an image is to be printed.
Newspaper	A newspaper-product
Notebook	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.
Pallet	Loaded pallet of boxes, cartons or Component resources.
Postcard	A card designed for sending a message by mail without an envelope.
Poster	A large printed picture.

Table A.69: Product Types (Sheet 3 of 3)

VALUE	DESCRIPTION
Proof	A representation that visualizes the intended output of page assembly, or the printing process. Proof SHOULD NOT be specified for a product as defined in ▶ Section 4.1.1 Product.
ResponseCard	A self mailer to respond to an offer.
Section	Main division of a book, such as a chapter, typically with a name or number.
SelfMailer	A document to be sent via the post without an additional envelope.
Spine	The bound edge of a book. Also, the portion of the cover that connects the front and back cover, wrapping the binding edge.
Stack	Stacked Component .
WrapAroundCover	A single cover sheet containing the front cover, spine and back cover.

A.3.18 Spine Operations

Table A.70: Spine Operations

VALUE	DESCRIPTION
Brushing	Brushes away dust from the spine to improve the binding quality.
FiberRoughing	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.
Leveling	After milling the spine, any uneven areas are leveled to achieve an even surface.
Milling	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.
Notching	This gives a clamping effect on the spine that is desirable for some products.
Sanding	Used for voluminous book papers.
Sealing	Apply heat to a spine of a book that contains signatures that have been prepared by ThreadSealing .
Shredding	Produces a relatively smooth surface. Further operations like "Notching", "Leveling", "FiberRoughing", "Sanding" or "Brushing" are necessary.

A.3.19 Status Details

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/@Status`, `QueueEntry/@Status` or device specific state `DeviceInfo/@Status`. Localized user data SHOULD be specified in `@DescriptiveName` or `Comment` elements.

A.3.19.1 Status Details for Generic Devices

Table A.71: Status Details Mapping for Generic Devices (Sheet 1 of 4)

STATUS DETAILS	NODEINFO/ @STATUS ^A	DEVICEINFO/ @STATUS	DESCRIPTION
AbortedBySystem	Aborted	Stopped	The job is being or has been aborted by the device.

Table A.71: Status Details Mapping for Generic Devices (Sheet 2 of 4)

STATUS DETAILS	NODEINFO/ @STATUS ^A	DEVICEINFO/ @STATUS	DESCRIPTION
BreakDown	Stopped	Offline	Breakdown of the device, repair needed.
Calibrating	Setup	Production	The device is calibrating, either manually or automatically.
ControlDeferred	-	Offline	The machine is not accessible by the device.
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	Production	Production of products in progress, good copy counter is on, waste copy counter is off.
Idling	Stopped	Production	Device is running, but no products are produced or consumed. Good and waste copy counters are off.
InputTrayMissing	Stopped	Stopped	One or more input trays are not in the device.
InterlockOpen	Stopped	Stopped	One or more interlock devices on the printer are unlocked.
IterationPaused	Suspended	Production	At least one iteration cycle has completed but additional iteration cycles MAY still occur.
JobCanceledByOperator	Aborted	Production	The job was canceled by the device operator using ModifyQueueEntry/ModifyQueueEntryParams/@Operation="Abort" , or means local to the device.
JobCanceledByUser	Aborted	Production	The job was canceled by the owner of the job using ModifyQueueEntry/ModifyQueueEntryParams/@Operation="Abort" .
JobCompletedSuccessfully	Completed	Production	The job completed successfully.
JobCompletedWithErrors	Completed	Production	The job completed with errors (and possibly warnings too).
JobCompletedWithWarnings	Completed	Production	The job completed with warnings.
JobHeld	Waiting	Production	The device held the job that had been waiting (by performing a ModifyQueueEntry/ModifyQueueEntryParams/@Operation="Hold" request on a waiting QueueEntry).
JobHeldOnCreate	Waiting	Production	The job was submitted to the queue with the QueueSubmissionParams/@Activation="Held" .
JobIncoming	Waiting	Production	The device is retrieving/accepting document data.
JobMissResources	Waiting InProgress	Stopped	When @Status is "InProgress" or "Waiting", QueueEntry waits for resources to become available to process further.

Table A.71: Status Details Mapping for Generic Devices (Sheet 3 of 4)

STATUS DETAILS	NODEINFO/ @STATUS ^A	DEVICEINFO/ @STATUS	DESCRIPTION
JobReadyForStart	Waiting InProgress	Stopped	When is @Status "InProgress" or "Waiting", QueueEntry is ready and waits for (manual) start event to process further.
JobResuming	Waiting	Production	The device is in the process of moving the job from a suspended condition to a candidate for processing (ModifyQueueEntry/ModifyQueueEntryParams/@Operation="Resume").
JobScheduling	Waiting	Production	The device is scheduling the job for processing.
JobStreaming	InProgress	Production	Same as "JobIncoming" 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).
JobSuspended	Suspended	Production	The device suspended the job that had been processing (e.g., by performing a ModifyQueueEntry/ModifyQueueEntryParams/@Operation="Suspend" request on a running QueueEntry) and other jobs can be processed by the device.
JobSuspending	InProgress	Production	The device is in the process of moving the job from a processing condition to a suspended condition where other jobs can be processed.
JobUserInputRequired	Waiting InProgress	Stopped	When @Status is "Waiting" or "InProgress", QueueEntry is not producible and waits for user input required to process further, e.g. missing parameters, decisions, etc.
Maintenance	Stopped	Stopped	General maintenance of the device.
MissResources	Stopped	Stopped	Production has been stopped because resources are missing or unavailable. Waits for new resources; subcategory of "Pause".
MovingToPaused	InProgress	Production	The device has been paused, but the machine(s) are taking an appreciable time to stop.
OutputAreaFull	Stopped	Stopped	One or more output areas are full (e.g., tray, stacker, collator).
OutputTrayMissing	Stopped	Stopped	One or more output trays are not in the device.
PaperJam	Stopped	Stopped	Media jam in the device; subcategory of "Failure".
Pause	Stopped	Stopped	Machine paused; restart is possible.
PendingReturn	Cleanup	Production	When @Status is "Cleanup", QueueEntry is currently returning.
ProcessingToStopPoint	InProgress	Production	The requester has issued an ModifyQueueEntry/ModifyQueueEntryParams/@Operation="Abort" 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.
QueuedToRun	Waiting	Stopped	When @Status is "Waiting", QueueEntry is queued to run and waits for device to become available (idle) to process further.
Repair	Stopped	Offline	The device is being repaired after a break down.

Table A.71: Status Details Mapping for Generic Devices (Sheet 4 of 4)

STATUS DETAILS	NODEINFO/ @STATUS ^A	DEVICEINFO/ @STATUS	DESCRIPTION
Running	InProgress	Production	When @Status is "InProgress", QueueEntry is processing.
ShutDown	Stopped	Offline	Machine stopped (can be switched off), restart requires a run up.
SizeChange	Setup	Production	Changing setup for media size.
StandBy	-	Idle	The device has been switched into power save mode and is still accepting new jobs.
StandBy	-	Offline	The device has been switched into power saving mode and cannot process jobs without prior intervention such as CommandWakeUp .
WaitForApproval	Stopped	Stopped	Production has been stopped because a necessary approval is still missing; subcategory of "Pause".
WarmingUp	Setup	Production	Device is warming up after power up or power saver mode wake-up.
Waste	InProgress	Production	Production of products in progress, good copy counter is off, waste copy counter is on.
WasteFull	Stopped	Stopped	The device waste receptacle is full.

a. The column **NodeInfo/@Status** also applies to **JobPhase/@Status** and **QueueEntry/@Status**.

A.3.19.2 Status Details for Printing Devices

Table A.72: Status Details Mapping for Printing Devices (Sheet 1 of 2)

STATUS DETAILS	NODEINFO/ @STATUS	DEVICEINFO/ @STATUS	DESCRIPTION
BlanketChange	Stopped	Stopped	Changing of blankets; subcategory of "Maintenance".
BlanketWash	Cleanup	Production	Washing of the blanket; subcategory of "WashUp".
CleaningInkFountain	Cleanup	Production	Cleaning of the ink fountain; subcategory of "WashUp".
CylinderWash	Cleanup	Production	Washing of impression cylinders; subcategory of "WashUp".
DampeningRollerWash	Cleanup	Production	Washing of the dampening roller; subcategory of "WashUp".
FormChange	Setup	Production	In conventional printing, changing of plates.
InkRollerWash	Cleanup	Production	Washing of the inking roller; subcategory of "WashUp".
PlateWash	Cleanup	Production	Washing of the plate; subcategory of "WashUp".
Processing	InProgress	Production	Other productive processing (RIP, etc.) is taking place but no final output is being produced. All input data has arrived (not "JobStreaming"/"InProgress" nor "JobIncoming"/"Waiting").
SleeveChange	Stopped	Stopped	Changing of sleeves; subcategory of "Maintenance".
WaitingForMarker	Suspended	Production	Processing is automatically suspended by the device because it is waiting behind other jobs in the marker module. The device will resume processing when a marker module becomes available.

Table A.72: Status Details Mapping for Printing Devices (Sheet 2 of 2)

STATUS DETAILS	NODEINFO/ @STATUS	DEVICEINFO/ @STATUS	DESCRIPTION
WashUp	Cleanup	Production	Machine is washed before, during or after production.

A.3.19.3 Status Details for Postpress Devices

Table A.73: Status Details Mapping for Postpress Devices

STATUS DETAILS	NODEINFO/ @STATUS	DEVICEINFO/ @STATUS	DESCRIPTION
BadFeed	Stopped	Stopped	Bad feed on a feeder; subcategory of "Failure".
BadTrim	Stopped	Stopped	Bad trimmed components; subcategory of "Failure".
DoubleFeed	Stopped	Stopped	Double feeds on a feeder; subcategory of "Failure".
IncorrectComponent	Stopped	Stopped	Incorrect components on a feeder; subcategory of "Failure".
IncorrectThickness	Stopped	Stopped	Incorrect thickness of components; subcategory of "Failure".
ObliqueSheet	Stopped	Stopped	Oblique sheets on components; subcategory of "Failure". Oblique sheets are sheets or signatures that are not properly aligned within a pile (e.g., on a gathering or collecting chain).

A.3.20 Texture

The following table defines a list of values that are valid for indicating the intended texture of the item to be used. This is typically the media or substrate.

Note: Values of the form IPP:xxx are provided for mapping to PWG Print Job Ticket. See ▶ [PWGMAP].

Table A.74: Texture

VALUE	DESCRIPTION
Antique	Rougher than vellum surface.
Calendared	Extra smooth or polished, uncoated paper.
Glossy	Glossy media.
IPP:Course	Generic value for coarse finish.
IPP:Fine	Generic value for fine finish.
IPP:Medium	Generic value for finish that is neither IPP:Fine nor IPP:Course.
Linen	Texture of coarse woven cloth.
Matte	Matte media.
Smooth	Generic term for smooth paper.
Stipple	Fine pebble finish.
Vellum	Slightly rough surface.

A.3.21 Units

The following defines a list of values that are valid for indicating the unit of a measurement quantity.

Note: The values in the following table are ordered alphabetically by measurement type.

Table A.75: Units

VALUE	MEASUREMENT	UNIT	DESCRIPTION
degree	Angle	degree°	An angle in degrees.
m2	Area	m ²	Used for media (e.g., in wide format printing).
count	Countable Objects	1	Countable objects, such as sheets, MAY be specified as “count”.
pt	Length	point (1/72 inch)	Used for all except microscopic lengths (see below).
um	Length	micron (μ)	Used for microscopic lengths — where used (instead of points) it will be explicitly stated in the definition of the item. See Media/@Thickness .
lpi	Line Screen	lpi	The lines per inch (lpi) for conventionally screened halftone, screened gray scale and screened monotone bitmap images.
gsm	Paper weight	g/m ²	Paper weight SHALL be provided in grams per square meter. See ▶ Appendix C Media Weight for details of calculating paper weights that are not in g/m ² .
kWh	Power (electrical)	kilowatt hour	Used to measure consumption of electricity. Note: Current power consumption (kW) MAY be provided in a ResourceInfo as ‘rate of consumption’ of electric power, i.e. kWh/h=kW.
dpi	Resolution	dpi	The dots per inch (dpi) for print output and bitmap image (TIFF, BMP, etc.) file resolution.
ppi	Screen Resolution	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.
spi	Spot Resolution	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) are not necessarily equivalent.
C	Temperature	°C (Celsius)	The temperature in degrees centigrade.
m3	Volume (gas)	m ³ (cubic meter)	Used to measure consumption of gas.
l	Volume (liquid)	liter	The volume in liters.
g	Weight	gram	The weight in grams.

Appendix B

B Return Values

The following list defines the standard return code for messaging. Return code values are integers. Error values below 100 are reserved for protocol errors. Error values above 100 are used for device and controller errors, while those higher than 200 refer to job and pipe specific errors.

Table B.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.
14	Subscription denied. An identical subscription already exists.
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	Cannot execute 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. Subsequent changes are 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.

Table B.1: Return codes for XJMF (Sheet 2 of 2)

RETURNCODE	DESCRIPTION
111	Subscription request denied.
112	Queue request failed because the queue is closed or blocked and did not accept new entries.
113	Queue entry is already in the resulting status.
114	QueueEntry/@Status is already "Completed" or "Aborted" and therefore does not accept changes.
115	Queue entry is not running.
116	Queue entry already exists. Used when a QueueEntry with identical @JobID , @JobPartID and Part already exists.
120	Cannot access referenced URL or URI reference cannot be resolved. Used when a referenced entity, e.g. an XJDF in a SubmitQueueEntry , cannot be found.
121	Unknown @DeviceID . No device is known with the @DeviceID specified.
130	Ganging is not supported. A gang job has been submitted to a queue that does not support ganging.
131	GangName not known. A job has been submitted with an unknown GangName .
140	CommandResource rejected.
141	CommandResource partially rejected.
200 – ...	Job and pipe specific errors.
200	Invalid resources.
201	Insufficient resource parameters.
202	PipeID unknown.
203	(Unused).
204	(Unused).
205	Pipe request failed because the pipe is closed and does not accept new requests.

Appendix C

C Media Weight

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.

C.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/m^2) 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/m^2 equals a basis weight of 67.5lb. Following is the English/grammage conversion formula:

Basis Weight (lb.) x (1406.5 / Square inches in basic size) = 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 the North American weight is multiplied by to calculate @Weight for various stock types. Stock type is specified in [Media/@StockType](#) or [MediaIntent /@StockType](#).

Table C.1: Conversion Factor from Basis Weight (lbs) to Weight (g/m^2)

STOCK TYPE	BASIS SIZE IN INCHES	CONVERSION FACTOR	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 C.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 C.2: Grammage Equivalents for Common (US) Basis Weights (Sheet 2 of 2)

GRAMMAGE (G/M ²)	BASIS WEIGHT	GRAMMAGE (G/M ²)	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

C.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 was originally published by EDS Inc., Editorial & Design Services; see ▶ [Japanese Paper Sizes]. For more help with grammage and basis weight conversion, see also ▶ [Grammage Conversion].

Following is the Japanese/grammage conversion formula:

$$\text{Basis Weight (kg) / Basic Size (m}^2\text{)} = \text{grams per square meter}$$

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 C.3: Japanese Media Weight (Sheet 1 of 2)

STOCK TYPE	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)
Aatoposutoshi アートポスト紙	180	-	125	-	209.3
	200	-	139	-	232.6
	220	-	153	-	255.0
Aatoshi アート紙	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

Table C.3: Japanese Media Weight (Sheet 2 of 2)

STOCK TYPE	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)
Chuushitsushi 中質紙	-	45	-	30	54.2
	-	55	-	36.5	66.3
Joushitsu 上質紙	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
Mashinkootoshi マシンコート紙	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

The following describes the five stock types in the above table:

- 上質紙 Joushitsu (“top-quality paper”) contains 100% chemical pulp
- 中質紙 Chuushitsu (“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

C.3 Paper Grade

▶ [ISO12647-2:2004] provides a rough classification of paper with 5 classes, which is generally referred to as paper grade. ▶ [ISO12647-2:2013] was updated in 2013, and a new set of 8 standard papers was defined that are more appropriate for paper types that are used today. The following table provides a rough and non-normative translation between the two standard sets:

Table C.4: Translation of Paper grades between [ISO12647-2:2004] and [ISO12647-2:2013]

ISO12647-2:2013		PRESS	ISO12647-2:2004	
ID	TYPE		GRADE	TYPE
PS1	Premium Coated	Sheet	1	Gloss coated
PS2	Improved Coated	Web	3	Gloss coated, web
PS3	Standard Coated Glossy	Web	3	Gloss coated, web
PS4	Standard Coated Matte	Web	2	Matte coated
PS5	Wood free Uncoated	Sheet	4	Uncoated white

Table C.4: Translation of Paper grades between [ISO12647-2:2004] and [ISO12647-2:2013]

ISO12647-2:2013		PRESS	ISO12647-2:2004	
ID	TYPE		GRADE	TYPE
PS6	Super Calendered	Web	4	Uncoated white
PS7	Improved Uncoated	Web	4	Uncoated white
PS8	Standard Uncoated	Web	4/5	Uncoated white/yellowish

Appendix D

D Media Size

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 points.

D.1 Architectural Paper Sizes

Table D.1: Architectural Paper Sizes

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
ArchA	648 x 864	228.6 x 304.8	9 x 12
ArchB	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 1219.2	36 x 48
ArchE1	2160 x 3024	762.0 x 1066.8	30 x 42
ArchE2	1872 x 2736	660.4 x 965.2	26 x 38
ArchE3	1944 x 2808	685.8 x 990.6	27 x 39

D.2 Business Card Sizes

Table D.2: Business Card Sizes

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
BusinessCard_Japan	156 x 258	55 x 91	2.2 x 3.6
BusinessCard_UK	156 x 241	55 x 85	2.2 x 3.3
BusinessCard_US	145 x 252	51 x 89	2.0 x 3.5

D.3 International A Paper Sizes

These sizes are defined by ISO standards, including ▶ [ISO216:2007] and by JIS standards ▶ [JIS P0138] except where noted.

Table D.3: International A Paper Sizes (Sheet 1 of 2)

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
A0	2384 x 3370	841 x 1189	33.11 x 46.81
A1	1684 x 2384	594 x 841	23.39 x 33.11
A2	1191 x 1684	420 x 594	16.54 x 23.39

Table D.3: International A Paper Sizes (Sheet 2 of 2)

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
A3	842 x 1191	297 x 420	11.69 x 16.54
A3Extra ^a	913 x 1262	322 x 445	12.67 x 17.52
A4	595 x 842	210 x 297	8.27 x 11.69
A4Extra ^a	667 x 914	235 x 322	9.27 x 12.69
A4Plus ^a	595 x 936	210 x 330	8.27 x 13.00
A4Tab ^a	638 x 842	225 x 297	8.86 x 11.69
A5	420 x 595	148 x 210	5.83 x 8.27
A5Extra ^a	492 x 668	174 x 235	6.85 x 9.25
A6	297 x 420	105 x 148	4.13 x 5.83
A7	210 x 297	74 x 105	2.91 x 4.13
A8	148 x 210	52 x 74	2.05 x 2.91
A9	105 x 148	37 x 52	1.46 x 2.05
A10	73 x 105	26 x 37	1.02 x 1.46

a. Non-standard ISO size variations.

D.4 International B Paper Sizes

These sizes are defined by ISO standards, including ▶ [ISO216:2007] and by JIS standards ▶ [JIS P0138].

Table D.4: International B Paper Sizes

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
B0	2920 x 4127	1030 x 1456	40.55 x 57.32
B1	2064 x 2920	728 x 1030	28.66 x 40.55
B2	1460 x 2064	515 x 728	20.28 x 28.66
B3	1032 x 1460	364 x 515	14.33 x 20.28
B4	729 x 1032	257 x 364	10.12 x 14.33
B5	516 x 729	182 x 257	7.17 x 10.12
B6	363 x 516	128 x 182	5.04 x 7.17
B7	258 x 363	91 x 128	3.58 x 5.04
B8	181 x 258	64 x 91	2.52 x 3.58
B9	127 x 181	45 x 64	1.77 x 2.52
B10	91 x 127	32 x 45	1.26 x 1.77

D.5 International C Envelope Sizes

These sizes are defined by ISO standards, including ▶ [ISO216:2007].

Table D.5: International C Envelope Sizes

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
C0	2599 x 3676	917 x 1297	36.1 x 51.1
C1	1837 x 2599	648 x 917	25.5 x 36.1
C2	1298 x 1837	458 x 648	18.0 x 25.5
C3	918 x 1298	324 x 458	12.8 x 18.0
C4	649 x 918	229 x 324	9.0 x 12.8
C5	459 x 649	162 x 229	6.4 x 9.0
C6	323 x 459	114 x 162	4.5 x 6.4
C7	230 x 323	81 x 114	3.2 x 4.5
C8	162 x 230	57 x 81	2.2 x 3.2
C9	113 x 162	40 x 57	1.6 x 2.2
C10	79 x 113	28 x 40	1.1 x 1.6

D.6 RA and SRA Paper Sizes

Table D.6: RA and SRA Paper Sizes

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
RA0	2438 x 3458	860 x 1220	33.9 x 48.0
RA1	1729 x 2438	610 x 860	24.0 x 33.9
RA2	1219 x 1729	430 x 610	16.9 x 24.0
RA3	865 x 1219	305 x 430	12.0 x 16.9
RA4	609 x 865	215 x 305	8.5 x 12.0
SRA0	2551 x 3628	900 x 1280	35.4 x 50.4
SRA1	1814 x 2551	640 x 900	25.2 x 35.4
SRA2	1276 x 1814	450 x 640	17.7 x 25.2
SRA3	907 x 1276	320 x 450	12.6 x 17.7
SRA4	638 x 907	225 x 320	8.9 x 12.6

D.7 US ANSI Paper Sizes

Table D.7: US ANSI Paper Sizes (Sheet 1 of 2)

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
AnsiA ^a	612 x 792	215.9 x 279.4	8.5 x 11
AnsiB ^b	792 x 1224	279.4 x 431.8	11 x 17

Table D.7: US ANSI Paper Sizes (Sheet 2 of 2)

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
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 1117.6	34 x 44

- a. Equivalent to US Letter.
- b. Equivalent to US Ledger & Tabloid.

D.8 US Paper Sizes

Table D.8: US Paper Sizes

MEDIA SIZE	SIZE IN POINTS	SIZE IN MILLIMETERS	SIZE IN INCHES
HalfLetter	396 x 612	139.7 x 215.9	5.5 x 8.5
Letter ^a	612 x 792	215.9 x 279.4	8.5 x 11
Legal	612 x 1008	215.9 x 355.6	8.5 x 14
JuniorLegal	360 x 576	127.0 x 203.2	5 x 8
LedgerTabloid ^b	792 x 1224	279.4 x 431.8	11 x 17

- a. Equivalent to ANSI A.
- b. Equivalent to ANSI B.

Appendix E

E String Generation

XJDF specifies a set of `@XXXFormat @XXXTemplate` pairs that allow dynamic generation of strings.

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 list of values selected from ▶ Table E.1 Predefined variables used in `@XXXTemplate`.

Table E.1: Predefined variables used in `@XXXTemplate` (Sheet 1 of 2)

NAME	'C' LANGUAGE DATATYPE	DESCRIPTION
<a partition key>	string	Any partition key that is an attribute of Part ; see ▶ Table 6.4 Part Element.
ActualAmount	float	Actual amount of the product that was produced.
Amount	float	Planned amount of the product that was produced.
CustomerID	string	CustomerInfo / <code>@CustomerID</code> .
CustomerName	string	Contact/Person of the customer contact. The sequence and selection of attributes is system dependent.
Date	string	Current date in ▶ [ISO8601:2004] format.
DeviceID	string	Device / <code>@DeviceID</code> of the device that produced the output.
DeviceName	string	Device / <code>@DescriptiveName</code> of the device that produced the output.
EndTime	string	Actual end time of the job.
Error	string	List of errors that occurred during the job. The formatting of errors is system dependent.
ErrorStats	string	Statistics on errors that happened during execution. The formatting of error statistics is system dependent.
ExposedMediaName	string	Resource / <code>@DescriptiveName</code> of the plate that is being imaged.
GeneralID:XXX	string	GeneralID / <code>@IDValue</code> of a GeneralID [<code>@IDUsage = "XXX"</code>]. For example if <code>@Format = "%i"</code> and <code>@Template = "GeneralID:foo"</code> then for: <code><GeneralID IDUsage="foo" IDValue="1"/></code> the extracted value is 1.
Generated	string	System generated string, for example a file name.
Input	string	Local file name of the input file. A value of "Input" SHALL NOT be specified for a FileSpec that describes an input file.
JobID	string	XJDF / <code>@JobID</code> of the job.
JobName	string	XJDF / <code>@DescriptiveName</code> of the job that is being processed.
JobPartID	string	XJDF / <code>@JobPartID</code> of the job.
MediaBrand	string	Resource / <code>@Brand</code> of the media that is being printed.
MoonPhase	string	Phase of the moon at the <code>@StartTime</code> of the job.
Operator	string	Contact/Person that describes the operator. The sequence and selection of attributes is system dependent.

Table E.1: Predefined variables used in @XXXTemplate (Sheet 2 of 2)

NAME	'C' LANGUAGE DATATYPE	DESCRIPTION
OperatorText	string	Text from the operator as defined in <code>Comment[@Type = "OperatorText"]</code> .
PressProfileName	string	The value of <code>ColorSpaceConversionParams/FileSpec/@UserFileName</code> of the <code>ColorSpaceConversion</code> process that is used for final output on the press.
PrintQuality	string	The value of <code>InterpretingParams/@PrintQuality</code> .
ProoferProfileName	string	The value of <code>ColorSpaceConversionParams/FileSpec/@UserFileName</code> of the <code>ColorSpaceConversion</code> process that is used for proofing.
Resolution	int	The value of <code>ObjectResolution/@Resolution</code> .
ResolutionX	int	The first (X) value of <code>ObjectResolution/@Resolution</code> .
ResolutionY	int	The first (Y) value of <code>ObjectResolution/@Resolution</code> .
ScreeningFamily	string	The value of <code>ScreeningParams/ScreenSelector/@ScreeningFamily</code> .
StartTime	string	Actual start time of the job.
Time	string	Current time in ▶ [ISO8601:2004] format.
TotalPagesInDoc	int	Value of <code>RunList/@NPage</code> of the current document.
Warning	string	Warnings that happened during the job. Warnings don't lose information in the resulting job, while errors do. The formatting of warnings is system dependent.

Example E.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/next/j001/m0000.pdf"
```

```
...
```

```
"file://myserver/next/j001/m2023.pdf"
```

```
<RunList>
  <FileSpec FileFormat="file://myserver/next/%s/m%4.i.pdf"
    FileTemplate="JobID DocIndex" MimeType="application/pdf"/>
</RunList>
```

F 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 ▶ Figure A-1: Fold catalog), which describes how to paginate single-sheet bindery signatures

F.1 How to interpret the diagrams

F.1.1 Legend

This appendix describes the structure and arrangement of bindery signatures into pagination schemes, which divide sheet surfaces into grids of rectangular areas to be filled by pages during the 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.

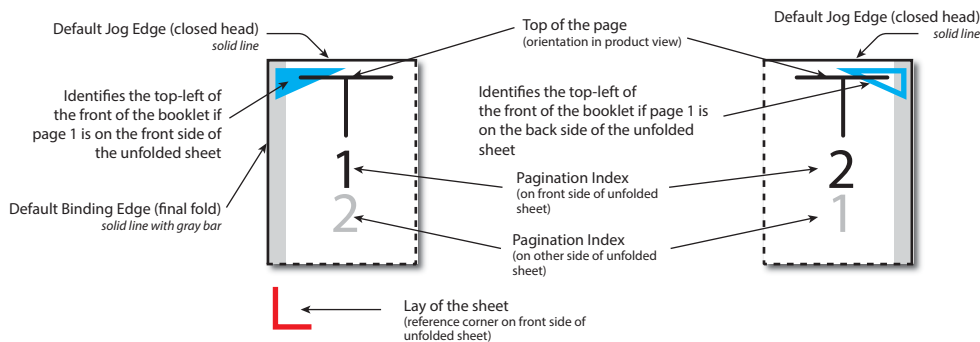
This appendix uses diagrams to describe the pagination schemes. Each diagram shows a side of an unfolded sheet, illustrating how it is divided into "signature cells". All cells are usually of the same size, allowing the entire sheet to be divided into equal portions, with each portion covering the whole 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.

Note: In contrast to the usual convention in this specification that all indices are 'zero' based, the page numbering in the diagrams in this appendix are 'one' based for readability.

The diagrams also show the pagination to be used when pages are flowed in reverse order because of different binding options, see ▶ Section F.1.3 Modifying the Pagination Schemes with BindingOrientation.

Figure F-1: Legend for interpreting diagrams



Folding sequences are described using the same notation found in ▶ Section A.3.7 Fold Catalog.

F.1.2 Meaning of a Pagination Scheme

The diagrams in ▶ Section F.2 Pagination Diagrams show the configuration of the page cells that occurs when the bindery signature is specified using [BinderySignature/@FoldCatalog](#).

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 with a booklet.

When multiple [BinderySignatures](#) are assembled together, the imposition indexes have to be translated into numbers referring to the list of source document pages. This is calculated using [Assembly/@Order](#) and, if specified, [AssemblySection/@BinderySignatureID](#).

The numbers and page orientation shown in the diagram correspond to the finished product view in the reader's perspective.

F.1.3 Modifying the Pagination Schemes with BindingOrientation

BinderySignature/**@BindingOrientation** MAY be set to indicate that the reference corner SHALL be displaced. This modifies the location of the spine, head, face and foot on the booklet before pagination is applied, e.g. for binding calendars or books to allow for a right to left reading order.

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.

F.1.4 Examples of applying BindingOrientation

The following examples describe the default orientation and pagination of **BinderySignature** depending on **@BindingOrientation**.

Note: The orientation of the final fold is defined in the production coordinate system. The binding side of the final product always defaults to left and is modified by **@BindingOrientation** regardless of whether the final fold in production is horizontal or vertical.

If the value of **@BindingOrientation** is one of the flip values, i.e. "Flip0", "Flip90" etc, then the implied page ordering of the **BinderySignature** is reversed, e.g. for right to left reading order.

F.1.4.1 Signature with Horizontal Final Folds

The examples below show how to read the diagrams after applying **@BindingOrientation**. Each diagram shows the pagination of the lay-side diagram defined for fold catalog "F8-7" for a given **@BindingOrientation**.

Table F.1: Original before transformation

@BindingOrientation	ILLUSTRATION
Rotate0	

Table F.2: Signatures with horizontal final folds

@BindingOrientation	ILLUSTRATION	@BindingOrientation	ILLUSTRATION
Rotate0		Rotate180	
Rotate90		Rotate270	
Flip0		Flip180	

Table F.2: Signatures with horizontal final folds

@BindingOrientation	ILLUSTRATION	@BindingOrientation	ILLUSTRATION
Flip90		Flip270	

F.1.4.2 Signature with Vertical Final Folds

The examples below show how to read the diagrams after applying @BindingOrientation. Each diagram is an interpretation of the lay-side diagram defined for fold catalog "F12-11" for a given @BindingOrientation.

Table F.3: Original before transformation

@BindingOrientation	ILLUSTRATION
Rotate0	

Table F.4: Signatures with vertical final folds

@BindingOrientation	ILLUSTRATION	@BindingOrientation	ILLUSTRATION
Rotate0		Rotate180	
Rotate90		Rotate270	
Flip0		Flip180	

Table F.4: Signatures with vertical final folds

@BindingOrientation	ILLUSTRATION	@BindingOrientation	ILLUSTRATION
Flip90		Flip270	

F.2 Pagination Diagrams

Table F.5: Pagination Diagrams (Sheet 1 of 12)

FOLD CATALOG	GRID SIZE	DESCRIPTION
FOLDING SEQUENCE		
F2-1	1 x 1	
(No folding sequence)		
F4-1	2 x 1	
$\uparrow^{1/2}$		
F4-2	2 x 1	
$\downarrow^{1/2}$		
F6-1	3 x 1	
$\uparrow^{1/3} \downarrow^{1/3}$		
F6-2	3 x 1	
$\downarrow^{1/3} \uparrow^{1/3}$		
F6-3	3 x 1	Unsupported (gatefold)
$\downarrow^{1/4} \uparrow^{1/2}$		
F6-4	3 x 1	
$\uparrow^{1/3} \uparrow^{1/3}$		
F6-5	3 x 1	
$\uparrow^{2/3} \downarrow^{1/3}$		
F6-6	3 x 1	Unsupported (multiple page sizes)
$\uparrow^{3/4} \downarrow^{1/4}$		

Table F.5: Pagination Diagrams (Sheet 2 of 12)

FOLD CATALOG		GRID SIZE	DESCRIPTION
FOLDING SEQUENCE			
F6-7	3 x 1		Unsupported (multiple page sizes)
$\uparrow^{1/4} \downarrow^{1/4}$			
F6-8	3 x 1		
$\uparrow^{2/3} \uparrow^{1/3}$			
F8-1	4 x 1		
$\uparrow^{1/2} \uparrow^{1/4}$			
F8-2	4 x 1		
$\uparrow^{1/2} \downarrow^{1/4}$			
F8-3	4 x 1		
$\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4}$			
F8-4	4 x 1		
$\uparrow^{1/4} \uparrow^{1/2} \downarrow^{1/4}$			
F8-5	4 x 1		
$\uparrow^{1/4} \uparrow^{1/4} \uparrow^{1/4}$			
F8-6	4 x 1		
$\uparrow^{3/4} \downarrow^{1/4} \downarrow^{1/4}$			
F8-7	2 x 2		
$\uparrow^{1/2} + \uparrow^{1/2}$			
F10-1	5 x 1		
$\uparrow^{1/5} \downarrow^{1/5} \uparrow^{1/5} \uparrow^{1/5}$			
F10-2	5 x 1		
$\uparrow^{4/5} \downarrow^{1/5} \downarrow^{1/5} \downarrow^{1/5}$			

Table F.5: Pagination Diagrams (Sheet 3 of 12)

FOLD CATALOG		GRID SIZE	DESCRIPTION
FOLDING SEQUENCE			
F10-3	5 x 1	$\uparrow^{2/5} \downarrow^{2/5} \uparrow^{1/5}$	
F12-1	6 x 1	$\uparrow^{1/3} \downarrow^{1/3} \uparrow^{1/6}$	
F12-2	6 x 1	$\uparrow^{1/3} \uparrow^{1/3} \downarrow^{1/6}$	
F12-3	6 x 1	$\uparrow^{1/2} \downarrow^{1/6} \uparrow^{1/6}$	
F12-4	6 x 1	$\uparrow^{1/2} \downarrow^{1/6} \downarrow^{1/6}$	
F12-5	6 x 1	$\uparrow^{1/2} \downarrow^{1/3} \uparrow^{1/6}$	
F12-6	6 x 1	$\uparrow^{1/6} \downarrow^{1/6} \uparrow^{1/6} \downarrow^{1/6} \uparrow^{1/6}$	
F12-7	3 x 2	$\uparrow^{1/3} \downarrow^{1/3} + \uparrow^{1/2}$	
F12-8	3 x 2	$\uparrow^{2/3} \uparrow^{1/3} + \uparrow^{1/2}$	
F12-9	3 x 2	$\uparrow^{1/3} \uparrow^{1/3} + \uparrow^{1/2}$	

Table F.5: Pagination Diagrams (Sheet 4 of 12)

FOLD CATALOG	GRID SIZE	DESCRIPTION
FOLDING SEQUENCE		
F12-10	3 x 2	
$\begin{matrix} \uparrow^{2/3} \downarrow^{1/3} \\ + \\ \uparrow^{1/2} \end{matrix}$		
F12-11	3 x 2	
$\begin{matrix} \uparrow^{1/3} \\ + \\ \uparrow^{1/2} \\ + \\ \uparrow^{1/3} \end{matrix}$		
F12-12	2 x 3	
$\begin{matrix} \uparrow^{1/2} \\ + \\ \uparrow^{2/3} \downarrow^{1/3} \end{matrix}$		
F12-13	2 x 3	
$\begin{matrix} \uparrow^{1/2} \\ + \\ \uparrow^{1/3} \uparrow^{1/3} \end{matrix}$		
F12-14	2 x 3	
$\begin{matrix} \uparrow^{1/2} \\ + \\ \uparrow^{1/3} \downarrow^{1/3} \end{matrix}$		
F14-1	7 x 1	
$\uparrow^{1/7} \downarrow^{1/7} \uparrow^{1/7} \downarrow^{1/7} \uparrow^{1/7} \uparrow^{1/7}$		
F16-1	8 x 1	
$\uparrow^{1/2} \downarrow^{1/4} \uparrow^{1/8}$		
F16-2	8 x 1	
$\uparrow^{1/2} \downarrow^{1/4} \downarrow^{1/8}$		

Table F.5: Pagination Diagrams (Sheet 5 of 12)

FOLD CATALOG		GRID SIZE	DESCRIPTION
FOLDING SEQUENCE			
F16-3	8 x 1	$\uparrow^{1/2} \uparrow^{1/4} \downarrow^{1/8}$	
F16-4	8 x 1	$\uparrow^{1/2} \uparrow^{1/4} \uparrow^{1/8}$	
F16-5	8 x 1	$\downarrow^{1/8} \uparrow^{1/8} \downarrow^{1/8} \uparrow^{1/8} \downarrow^{1/8} \uparrow^{1/8} \downarrow^{1/8}$	
F16-6	4 x 2	$\uparrow^{1/2} + \uparrow^{1/2} + \uparrow^{1/4}$	
F16-7	4 x 2	$\uparrow^{1/2} + \uparrow^{1/2} + \downarrow^{1/4}$	
F16-8	4 x 2	$\uparrow^{1/2} + \downarrow^{1/2} + \downarrow^{1/4}$	
F16-9	4 x 2	$\uparrow^{1/2} \downarrow^{1/4} + \uparrow^{1/2}$	
F16-10	4 x 2	$\uparrow^{1/2} \uparrow^{1/4} + \uparrow^{1/2}$	
F16-11	4 x 2	$\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4} + \uparrow^{1/2}$	

Table F.5: Pagination Diagrams (Sheet 6 of 12)

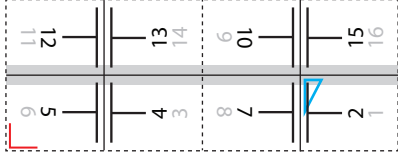
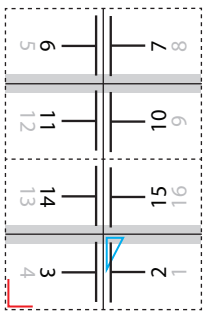
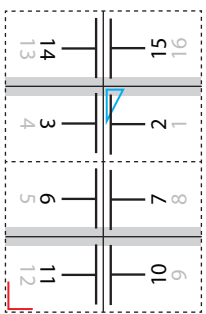
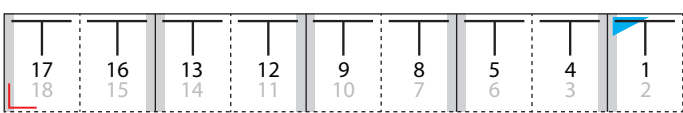
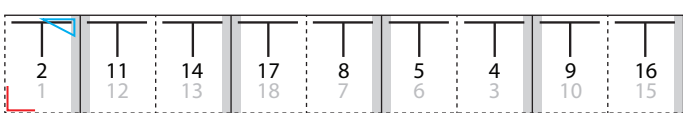
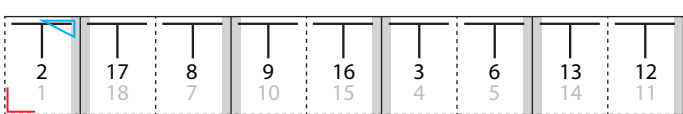
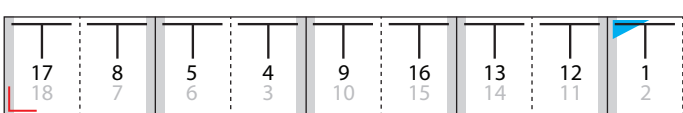
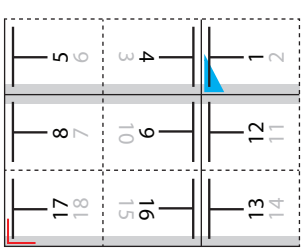
FOLD CATALOG	GRID SIZE	DESCRIPTION
FOLDING SEQUENCE		
F16-12	4 x 2	
F16-13	2 x 4	
F16-14	2 x 4	
F18-1	9 x 1	
F18-2	9 x 1	
F18-3	9 x 1	
F18-4	9 x 1	
F18-5	3 x 3	

Table F.5: Pagination Diagrams (Sheet 7 of 12)

FOLD CATALOG		GRID SIZE	DESCRIPTION
FOLDING SEQUENCE			
F18-6	3 x 3	$\begin{matrix} \uparrow^{1/3} \downarrow^{1/3} \\ + \\ \uparrow^{2/3} \downarrow^{1/3} \end{matrix}$	
F18-7	3 x 3	$\begin{matrix} \uparrow^{1/3} \uparrow^{1/3} \\ + \\ \uparrow^{1/3} \downarrow^{1/3} \end{matrix}$	
F18-8	3 x 3	$\begin{matrix} \uparrow^{1/3} \uparrow^{1/3} \\ + \\ \uparrow^{2/3} \downarrow^{1/3} \end{matrix}$	
F18-9	3 x 3	$\begin{matrix} \uparrow^{2/3} \uparrow^{1/3} \\ + \\ \uparrow^{2/3} \uparrow^{1/3} \end{matrix}$	
F20-1	5 x 2	$\begin{matrix} \uparrow^{2/5} \downarrow^{2/5} \uparrow^{1/5} \\ + \\ \uparrow^{1/2} \end{matrix}$	
F20-2	5 x 2	$\begin{matrix} \uparrow^{1/5} \downarrow^{1/5} \uparrow^{1/5} \downarrow^{1/5} \\ + \\ \uparrow^{1/2} \end{matrix}$	
F24-1	6 x 2	$\begin{matrix} \uparrow^{1/3} \downarrow^{1/3} \\ + \\ \uparrow^{1/2} \\ + \\ \uparrow^{1/6} \end{matrix}$	

Table F.5: Pagination Diagrams (Sheet 8 of 12)

FOLD CATALOG	GRID SIZE	DESCRIPTION
FOLDING SEQUENCE		
F24-2	6 x 2	
F24-3	6 x 2	
F24-4	6 x 2	
F24-5	6 x 2	
F24-6	6 x 2	
F24-7	6 x 2	
F24-8	3 x 4	

Table F.5: Pagination Diagrams (Sheet 9 of 12)

FOLD CATALOG		GRID SIZE	DESCRIPTION
FOLDING SEQUENCE			
F24-9	3 x 4	$\begin{matrix} \uparrow^{2/3} \uparrow^{1/3} \\ + \\ \uparrow^{1/2} \downarrow^{1/4} \end{matrix}$	
F24-10	3 x 4	$\begin{matrix} \uparrow^{1/3} \uparrow^{1/3} \\ + \\ \uparrow^{1/2} \downarrow^{1/4} \end{matrix}$	
F24-11	4 x 3	$\begin{matrix} \uparrow^{1/2} \\ + \\ \uparrow^{2/3} \downarrow^{1/3} \\ + \\ \uparrow^{1/4} \end{matrix}$	
F28-1	7 x 2	$\begin{matrix} \uparrow^{1/7} \downarrow^{1/7} \uparrow^{1/7} \downarrow^{1/7} \uparrow^{1/7} \downarrow^{1/7} \\ + \\ \uparrow^{1/2} \end{matrix}$	
F32-1	16 x 1	$\uparrow^{1/2} \downarrow^{1/4} \uparrow^{1/8} \downarrow^{1/16}$	
F32-2	8 x 2	$\begin{matrix} \uparrow^{1/2} \downarrow^{1/4} \\ + \\ \uparrow^{1/2} \\ + \\ \uparrow^{1/8} \end{matrix}$	

Table F.5: Pagination Diagrams (Sheet 10 of 12)

FOLD CATALOG	GRID SIZE	DESCRIPTION
FOLDING SEQUENCE		
F32-3	8 x 2	
F32-4	4 x 4	
F32-5	4 x 4	
F32-6	4 x 4	
F32-7	4 x 4	

Table F.5: Pagination Diagrams (Sheet 11 of 12)

FOLD CATALOG	GRID SIZE	DESCRIPTION
FOLDING SEQUENCE		
F32-8	4 x 4	
F32-9	4 x 4	
F36-1	9 x 2	
F36-2	6 x 3	
F40-1	5 x 4	

Table F.5: Pagination Diagrams (Sheet 12 of 12)

FOLD CATALOG	GRID SIZE	DESCRIPTION
FOLDING SEQUENCE		
F48-1	6 x 4	<p> $\uparrow^{1/3} \downarrow^{1/3}$ $+$ $\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4}$ $+$ $\uparrow^{1/6}$ </p>
F48-2	4 x 6	<p> $\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4}$ $+$ $\uparrow^{1/3} \downarrow^{1/3} \uparrow^{1/6}$ </p>
F64-1	8 x 4	<p> $\uparrow^{1/2}$ $+$ $\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4}$ $+$ $\uparrow^{1/4} \downarrow^{1/8}$ </p>
F64-2	8 x 4	<p> $\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4}$ $+$ $\uparrow^{1/4} \downarrow^{1/4} \uparrow^{1/4}$ $+$ $\uparrow^{1/8}$ </p>

Appendix G

G Hole Pattern Catalog

The following table defines the specifics of the predefined holes in [HoleMakingParams](#) and [HoleMakingIntent](#).

Notes:

- 1 All patterns are centered on the sheet along the reference edge.
- 2 The reference edge is always defined relative to a portrait orientation of the medium, regardless of the orientation of the printed image or processing path.
- 3 The pattern axis offset is always specified relative to the reference edge.
- 4 The default pattern axis offset is always specified in points.
- 5 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.
- 6 Single thumbcuts appear always in the center of the reference edge.
- 7 Oval-shaped holes sometimes actually look more like rectangular holes with rounded corners.
- 8 Generic hole types are dependent on the geographical area where the device is used.

Sources:

- 1 Printer Finishing MIB ▶ [RFC3806]
- 2 Files and folders; concepts ▶ [DIN 821-3]
- 3 Paper; Holes for general filing purposes; Specifications ▶ [ISO838]
- 4 Paper and board - Dimensions for standard punching of A4 and A5 paper ▶ [DIN 5005]

G.1 Naming Scheme

Table G.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

G.2 Ring Binding - Two Hole

Table G.2: Hole Details for R2 Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
R2-generic	Generic request of a 2-hole pattern	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	34.02 (≅ 12 mm)	Left	See note (8)	N/A
R2m-DIN	DIN 2-hole MIB: 6 = two-HoleDIN and 10 = twoHole-Metric	●	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 (≅ 11 mm)	Left	A4 and A5	[DIN 5005] [DIN 821-3]
R2m-ISO	ISO 2-hole MIB: 6 = two-HoleDIN and 10 = twoHole-Metric	●	6 ± 0.5 mm	80 ± 0.5 mm	12 ± 1 mm Australian Standard AS P5-1969: 10 ± 1 mm	34.02 (≅ 12 mm)	Left	Also used in Japan	[ISO838]
R2m-MIB	Printer Finishing MIB twoHoleDIN and twoHole-Metric	●	5-8 mm	80 ± 0.5 mm	4.5 - 13 mm	31.18 (≅ 11 mm)	Left		[RFC3806]
R2i-US-a	US 2-hole, Variant A MIB: 4 = two-HoleUSTop and 12 = twoHole-USSide	●	0.2 - 0.32"	2.75"	0.18 - 0.51"	29.25 (≅ 13/32")	Left for letter Top for ledger		[RFC3806]
R2i-US-b	US 2-hole, Variant B	●	0.2 - 0.5" std: 5/16" typ: 1/4", 9/32", 11/32", 3/8", 13/32", 1/2"	6"	0.25" + 1/2 diameter Range: 6/16" - 1/2"	29.25 (≅ 13/32")	Left		

G.3 Ring Binding - Three Hole

Table G.3: Hole Details for R3 Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
R3-generic	Generic request of a 3-hole pattern.	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	29.25 (≅ 13/32")	Left	See note (8)	N/A

Table G.3: Hole Details for R3 Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
R3i-US	US 3-hole MIB: 5 = threeHoleUS	●	0.2 - 0.5" std: 5/16" typ: 1/4", 9/32", 11/ 32", 3/ 8", 13/ 32", 1/2"	4.25"	0.25" + 1/2 diameter range: 6/16" - 1/ 2"	29.25 (≅ 13/32")	Left		[RFC3806]

G.4 Ring Binding - Four Hole

Table G.4: Hole Details for R4 Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
R4-generic	Generic request of a 4-hole pattern.	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	31.18 (≅ 11 mm)	Left	See note (8)	N/A
R4m-DIN-A4	DIN 4-hole for A4	●	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] [DIN 821-3]
R4m-DIN-A5	DIN 4-hole for A5	●	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 5005]
R4m-swedish	Swedish 4-hole MIB: 11 = swedish4Hole	●	5 - 8 mm	21-70-21 mm	4.5 - 13 mm	31.18 (≅ 11 mm)	Left for A4 Top for A3	A4, A3	[RFC3806]
R4i-US	US 4-hole	●	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" + 1/2 diameter Range: 6/16" - 1/ 2"	29.25 (≅ 0.25" + 1/2 x 5/16" = 13/32")	Left		

G.5 RingBinding - Five Hole

Table G.5: Hole Details for R5 Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
R5-generic	Generic request of a 5-hole pattern.	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	29.25 ($\cong 13/32''$)	Left	See note (8)	N/A
R5i-US-a	US 5-hole, Variant A MIB: 13 = five-HoleUS	●	0.2 - 0.32"	2-2.25-2.25-2"	0.18 - 0.51"	29.25 ($\cong 13/32''$)	Left for letter Top for ledger		[RFC3806]
R5i-US-b	US 5-hole, Variant B	●	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"	0.25" + 1/2 diameter 0.375 - 0.5"	29.25 ($\cong 0.25'' + 1/2 \times 5/16'' = 13/32''$)	Left		
R5i-US-c	Combination of R2i-US-a and R3i-US	●	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" + 1/2 diameter 0.375 - 0.5"	29.25 ($\cong 0.25'' + 1/2 \times 5/16'' = 13/32''$)	Left		

G.6 Ring Binding - Six Hole

Table G.6: Hole Details for R6 Series (Sheet 1 of 2)

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
R6-generic	Generic request of a 6-hole pattern.	●	5 - 13 mm 0.2-0.51"	N/A	4.5 - 13 mm 0.18 - 0.51"	31.18 ($\cong 11 \text{ mm}$)	Left for A4/A5 Top for A3	See note (8)	N/A
R6m-4h2s	Norwegian 4-hole (round) mixed with 2 slots (rectangular) MIB: 16 = norwegian6Hole	H: ● 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 ($\cong 11 \text{ mm}$)	Left for A4 Top for A3		[RFC3806]

Table G.6: Hole Details for R6 Series (Sheet 2 of 2)

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
R6m-DIN-A5	DIN 6-hole for A5	●	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]




G.7 Ring Binding - Seven Hole

Table G.7: Hole Details for R7 Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
R7-generic	Generic request of a 7-hole pattern.	●	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 (8)	N/A
R7i-US-a	US 7-hole, Variant A MIB: 14 = sevenHoleUS	●	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		[RFC3806]
R7i-US-b	US 7-hole, Bell/AT&T Systems. Combination of R3i-US, R4i-US, R5i-US-b	●	0.2 - 0.5" std: 5/16" typ: 1/4", 9/32", 11/32", 3/8", 13/32", 1/2"	0.75-1.375-2.125-2.125-0.75"	0.25" + 1/2 diameter 0.375 - 0.5"	29.25 (≅ 0.25" + 1/2 x 5/16" = 13/32")	Left for letter Top for ledger		
R7i-US-c	US 7-hole, Variant C	●	0.2 - 0.5" std: 5/16" typ: 1/4", 9/32", 11/32", 3/8", 13/32", 1/2"	1.25-0.875-2.125-2.125-0.875-1.25"	0.25" + 1/2 diameter 0.375 - 0.5"	29.25 (≅ 13/32")	Left for letter Top for ledger		





G.8 Ring Binding - Eleven Hole

Table G.8: Hole Details for R11 Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
R11m-7h4s	7-hole (round) mixed with 4 slots (rectangular) MIB: 15 = mixed7H4S	H:  S:  	Holes: 5 - 8 mm Slots: 12 x 6 mm	7 holes/ 2slots Pattern: H-S-H-H-S-H-S-H-H-S-H 15-25-23-20-37-37-20-23-25-15 mm	4.5 - 13 mm	31.18 (≅ 11 mm)	Left for A4 Top for A3		[RFC3806]

G.9 Plastic Comb Binding

Table G.9: Hole Details for P Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
P16_9i-rect-0t	US spacing, no thumbcut MIB: 9 = nineteenHoleUS A4: 21 Hole Letter: 19 Hole	 	5/16" x 1/8" (8 x 3.2 mm)	9/16"	3/16"	13.54 (≅ 0.188")	Left		[RFC3806]
P12m-rect-0t	European spacing, no thumbcut	 	7 x 3 mm	12 mm	4.5 mm	12.76 (≅ 4.5 mm)	Left		

G.10 Wire Comb Binding

Wire comb binding uses twenty-three holes for pages of A4 size, and twenty-one holes for pages of letter size.

Table G.10: Hole Details for W Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
W2_1i-round-0t	2:1, round, no thumbcut MIB: 8 = twentyTwo-HoleUS A4: 23 Hole Letter: 21 Hole	●	0.2 - 0.32" std: 1/4" Europe typ: 6 or 6.4 mm	1/2"	3 mm + 1/2 diameter 0.318 - 0.438" Europe: 6 - 6.2 mm	17.50 (≅ 0.243")	Left		[RFC3806]
W2_1i-square-0t	2:1, square, no thumbcut A4: 23 Hole Letter: 21 Hole	■	0.2 - 0.32" std: 1/4" Europe typ: 6 or 6.4 mm	1/2"	3 mm + 1/2 diameter 0.318 - 0.438" Europe: 6 - 6.2 mm	17.50 (≅ 0.243")	Left		
W3_1i-square-0t	3:1, square, no thumbcuts A4: 34 Hole A5: 24 Hole Letter: 32 Hole	■	5/32 x 5/32" (4x4 mm)	1/3"	0.2"	14.40 (≅ 0.2")	Left		

G.11 Coil and Spiral Binding

Table G.11: Hole Details for C Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
C9.5m-round-0t	9.5 mm, round, no thumbcut MIB: 17 = metric26Hole and 18 = metric30Hole A4: 30 Hole A3: 30 Hole JIS: 26 Hole B4: 26 Hole B5: 26 Hole	●	5 - 8 mm	9.5 mm	4.5 - 13 mm	31.18 (≅ 11 mm)	Left for A4/JIS B5 Top for A3/JIS B4		[RFC3806]

G.12 Special Binding

Table G.12: Hole Details for S Series

HOLE PATTERN ID	DESCRIPTION	SHAPE	HOLE EXTENT	PATTERN GEOMETRY	PATTERN AXIS OFFSET	XJDF DEFAULT PATTERN AXIS OFFSET	EDGE	NOTE	SOURCE
S-generic	Generic request of a hole pattern with an arbitrary or unknown number of holes (e.g., an inline shotgun)	●	5 - 13 mm 0.2-0.51"	N/A	N/A	N/A	Any		N/A
S1-generic	Generic request of a hole pattern with 1 hole	●	5 - 13 mm 0.2-0.51"	N/A	N/A	N/A	Any		N/A

Appendix H

H 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 appendix section to find the full title, date, source and availability of all such references.

Table H.1: References (Sheet 1 of 8)

TERM	DEFINITION
[Braille ASCII]	<i>Six dot Braille representation of the ASCII character set</i> Date: - Available at: https://en.wikipedia.org/wiki/Braille_ASCII
[Braille Unicode]	<i>Braille Patterns</i> Date: - Available at: https://en.wikipedia.org/wiki/Braille_Unicode_block
[BT.601-7]	<i>ITU Recommendation 601-7</i> <i>Studio encoding parameters of digital television for standard 4:3 and wide screen 16:9 aspect ratios.</i> Date: March 2011 Produced by: International Telecommunication Union - Radioncommunication Sector Available at: https://www.itu.int/rec/R-REC-BT.601/ International Telecommunication Union, General Secretariat — Sales Section, Place des Nations, CH-1211 Geneva 20 (Switzerland)
[CGATS.21]	<i>CGATS.21-1-2013 and CGATS.21-2-2013</i> <i>Graphic technology - Printing from digital data across multiple technologies Parts 1 & 2</i> Date: 2013 Produced by: Committee for Graphic Arts Technologies Standards (NPES serves as the American National Standards Institute (ANSI) secretariat to CGATS) Available at: http://www.npes.org/programs/standardsworkroom.aspx
[Characterization Data]	<i>CMYK Characterization Data</i> <i>Registered CMYK characterization data sets for standard printing processes.</i> Date: - Produced by: International Color Consortium Available at: http://www.color.org/chardata/drsection1.xalter
[CIE 015:2004]	<i>CIE 15:2004</i> <i>Colorimetry, 3rd Edition.</i> Date: 2004 Produced by: Commission Internationale de l'Eclairage International (CIE) Available at: http://www.cie.co.at/publications/colorimetry-3rd-edition
[CIP3 - PPF]	<i>CIP3 Print Production Format (PPF)</i> Date: 1 June 1998 Version: CIP3 PPF Specification 3.0 Produced by: CIP4 Organization Available at: https://confluence.cip4.org/display/PUB/PPF
[Color Names]	<i>Recognized color keyword names</i> Date: 16 August 2011 Version: SVG (1.1) Second Edition Produced by: World Wide Web Consortium (W3C) Available at: https://www.w3.org/TR/SVG/types.html#ColorKeywords

Table H.1: References (Sheet 2 of 8)

TERM	DEFINITION
[ColorPS]	<p>Color Separation Conventions for PostScript Language Programs Technical Note #5044 Date: 24 May 1996 Produced by: Adobe Systems Inc. Available at: https://www.adobe.com/content/dam/acom/en/devnet/postscript/pdfs/5044.ColorSep_Conv.pdf</p>
[Corrugated Packaging]	<p>Corrugated packaging Date: - Produced by: Corrugated Packaging Alliance Available at: http://www.corrugated.org/</p>
[DDES3]	<p>Graphic technology – Prepress digital data exchange – Diecutting data (DDES3) Date: Revised 2017 Produced by: ANSI® IT8.6-2017 Available at: http://webstore.ansi.org.</p>
[DIN 15146-4]	<p>Pallets; timber four-way-flat pallets; 800mm x 600mm Date: Revised 1991 Produced by: DIN Standards Committee Packaging Available at: https://www.din.de/en.</p>
[DIN 5005]	<p>Paper and board – Sheets for loose leaf binders – Dimensions for standard punching for paper sizes A4 and A5 Date: Edition 2008-2 Produced by: DIN Standards Committee Paper, board and pulps Available at: https://www.din.de/en.</p>
[DIN 821-3]	<p>Files and folders; concepts Date: Edition 1991-03 Produced by: DIN Standards Committee Timber and Furniture Available at: https://www.din.de/en.</p>
[DIN EN 13698-1]	<p>Pallet production specification – Part 1: Construction specification for 800mm x 1200mm flat wooden pallets Date: Edition 2004-01 Produced by: DIN Standards Committee Packaging Available at: https://www.din.de/en.</p>
[DIN EN 13698-2]	<p>Pallet production specification – Part 2: Construction specification for 1000mm x 1200mm flat wooden pallets Date: Revised 2009-10 Produced by: DIN Standards Committee Packaging Available at: https://www.din.de/en.</p>
[ECMA]	<p>ECMA Code of Folding Carton Design Styles Date: - Produced by: European Carton Makers Association Available at: https://www.ecma.org/publications/ecma-code-of-folding-carton-design-styles.html</p>
[FEFCO]	<p>FEFCO European Federation of Corrugated Board Manufacturers Date: - Produced by: European Federation of Corrugated Board Manufacturers Available at: http://www.fefco.org</p>
[FINAT]	<p>Trade Association for the Self-Adhesive and Labeling Industry Web site: http://www.finat.com</p>

Table H.1: References (Sheet 3 of 8)

TERM	DEFINITION
[FOGRA]	<p>Characterization data for standardized printing conditions Various Standards, e.g. FOGRA39, FOGRA47 etc</p> <p>Date: N/A Produced by: Fogra Research Institute for Media Technologies Available at: https://www.fogra.org/</p>
[Grammage Conversion]	<p>Basis Weight and Grammage Conversion Tables.</p> <p>Date: - Produced by: EDS Inc. Editorial & Design Services Available at: http://www.edsebooks.com/paper/grammage.html</p>
[IANA-character sets]	<p>IANA Registry of Character Set names</p> <p>Available at: https://www.iana.org/assignments/character-sets/character-sets.xhtml</p>
[IANA-mt]	<p>IANA Registry of MIME Media Types</p> <p>Date: 2018-02-09 Available at: http://www.iana.org/assignments/media-types</p>
[ICC.1]	<p>Specification ICC.1:2010-12 (Profile version 4.3.0.0) Image technology colour management – Architecture, profile format, and data structure</p> <p>Date: 2010 Produced by: International Color Consortium (ICC) Available at: http://www.color.org/</p>
[IEEE754]	<p>IEEE 754-2008 Standard for Binary Floating-Point Arithmetic</p> <p>Date: 2008 Produced by: IEEE Available at: http://grouper.ieee.org/groups/754/</p>
[ISO/IEC 15948:2004]	<p>ISO/IEC 15948:2004 Information technology -- Computer graphics and image processing -- Portable Network Graphics (PNG): Functional specification Second edition</p> <p>Date: 2004 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO/NP 3166-1:2013]	<p>ISO/NP 3166-1:2013 Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes.</p> <p>Date: 2013 Produced by: ISO Central Secretariat Available at: https://www.iso.org/store.html</p>
[ISO12647-2:2004]	<p>ISO 12647-2:2004 Graphic technology – Process control for the production of half-tone colour separations, proof and production prints – Part 2: Offset lithographic processes.</p> <p>Date: 2004 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO12647-2:2013]	<p>ISO 12647-2:2013 Graphic technology – Process control for the production of half-tone colour separations, proof and production prints – Part 2: Offset lithographic processes.</p> <p>Date: 2013 Produced by: ISO Available at: https://www.iso.org/store.html</p>

Table H.1: References (Sheet 4 of 8)

TERM	DEFINITION
[ISO13655:2017]	<p>ISO 13655:2017 <i>Graphic technology -- Spectral measurement and colorimetric computation for graphic arts images.</i> Date: 2017 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO15930-1:2001]	<p>ISO 15930-1:2001 <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> Date: 2001 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO15930-3:2002]	<p>ISO 15930-3:2002 <i>Graphic technology — Prepress digital data exchange — Use of PDF — Part 3: Complete exchange suitable for colour-managed workflows (PDF/X-3).</i> Date: 2002 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO15930-7:2010]	<p>ISO 15930-7:2010 <i>Graphic technology — Prepress digital data exchange — Use of PDF — Part 7: Complete exchange of printing data (PDF/X-4) and partial exchange of printing data with external profile reference (PDF/X-4p) using PDF 1.6.</i> Date: 2010 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO15930-8:2010]	<p>ISO 15930-8:2010 <i>Graphic technology — Prepress digital data exchange using PDF — Part 8: Partial exchange of printing data using PDF 1.6 (PDF/X-5).</i> Date: 2010 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO17972-1:2015]	<p>ISO 17972-1:2015 <i>Graphic technology -- Colour data exchange format -- Part 1: Relationship to CxF3 (CxF/X)</i> Date: 2015 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO19593-1:2016]	<p>ISO 19593-1:2016 <i>Graphic technology -- Use of PDF to associate processing steps and content data -- Part 1: Processing steps for packaging and labels</i> Date: 2016 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO2108:2017]	<p>ISO 2108:2017 <i>Information and documentation -- International standard book number (ISBN).</i> Date: 2017 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO216:2007]	<p>ISO 216:2007 <i>Writing paper and certain classes of printed matter -- Trimmed sizes -- A and B series.</i> Date: 1975 Produced by: ISO Available at: https://www.iso.org/store.html</p>

Table H.1: References (Sheet 5 of 8)

TERM	DEFINITION
[ISO2470-1:2016]	<p>ISO 2470-1:2016 <i>Paper, board and pulps -- Measurement of diffuse blue reflectance factor -- Part 1: Indoor daylight conditions (ISO brightness).</i> Date: 2016 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO2471:2008]	<p>ISO 2471:2008 <i>Paper and board—Determination of opacity (paper backing)—Diffuse reflectance method.</i> Date: 2008 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO8254-1:2009]	<p>ISO 8254-1:2009 <i>Paper and board -- Measurement of specular gloss -- Part 1: 75 degree gloss with a converging beam, TAPPI method</i> Date: 2009 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO838]	<p><i>Paper; Holes for general filing purposes; Specifications</i> Date: 1974 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[ISO8601:2004]	<p>ISO 8601:2004 <i>Data elements and interchange formats - Information interchange - Representation of dates and times.</i> Date: 2004 Produced by: ISO Available at: https://www.iso.org/store.html</p>
[Japanese Paper Sizes]	<p><i>Japanese Papers for Printing, Metric Measurements and Inch Equivalents.</i> Date: - Produced by: EDS Inc. Editorial & Design Services Available at: http://www.edsebooks.com/paper/jpaper.html</p>
[JDF15]	<p><i>Job Definition Format 1.5</i> Date: 2013 Produced by: International Cooperation for Integration of Processes in Prepress, Press and Postpress (CIP4) Available at: http://www.cip4.org</p>
[JIS P0138]	<p>JIS P 0138:1998 <i>Writing paper and certain classes of printed matter -- Trimmed sizes -- A and B series.</i> Date: 1998 Produced by: JIS Available at: https://webdesk.jsa.or.jp/books/W11M0010</p>
[K&R]	<p><i>The C Programming Language</i> , by Brian W. Kernighan and Dennis M. Ritchie Second Edition Date: March 22, 1988 Produced by: Prentice Hall Available at: Book only - ISBN 0131103628</p>
[PDF1.6]	<p><i>PDF reference : Adobe portable document format version 1.6</i> Date: November 2004 Produced by: Adobe Systems Incorporated Available at: https://www.adobe.com/devnet/pdf/pdf_reference_archive.html</p>

Table H.1: References (Sheet 6 of 8)

TERM	DEFINITION
[PostScript]	<p><i>PostScript Language Reference (Redbook)</i> <i>Third Edition</i></p> <p>Date: - Produced by: Adobe Systems Incorporated Available at: https://www.adobe.com/content/dam/acom/en/devnet/actionsript/articles/PLRM.pdf</p>
[PPD]	<p><i>Adobe PostScript Printer Description File Format Specification</i> <i>Version 4.3</i></p> <p>Date: 9 February 1996 Produced by: Adobe Systems Inc. Available at: https://www-cdf.fnal.gov/offline/PostScript/5003.PPD_Spec_v4.3.pdf</p>
[Pro Carton]	<p><i>European Association of Carton and Cartonboard manufacturers.</i></p> <p>Date: - Produced by: Pro Carton Available at: http://www.procarton.com/publications-news/publications/</p>
[PWGMAP]	<p><i>Mapping CIP4 JDF to PWG Print Job Ticket v1.0 (JDFMAP)</i></p> <p>Date: August 2017 Produced by: The Printer Working Group Available at: http://ftp.pwg.org/pub/pwg/informational/bp-smjdfmap10-20170828.pdf</p>
[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: http://www.rfc-editor.org/rfcsearch.html</p> <p>Note: This version is obsolete and has been replaced by: <i>RFC 4248, The telnet URI Scheme - 2005</i> <i>RFC 4266, The gopher URI Scheme - 2005</i></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: 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: http://www.rfc-editor.org/rfcsearch.html</p>
[RFC3066]	<p><i>RFC 3066</i> <i>Tags for the Identification of Languages, by H. Alvestrand.</i></p> <p>Date: January 2001 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: http://www.rfc-editor.org/rfcsearch.html</p>
[RFC3806]	<p><i>Printer Finishing MIB</i></p> <p>Date: June 2004 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: http://www.rfc-editor.org/rfcsearch.html</p>

Table H.1: References (Sheet 7 of 8)

TERM	DEFINITION
[RFC3966]	<p>RFC 3966 <i>The tel URI for Telephone Numbers</i> by H. Schulzrinne Date: December 2004 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: http://www.rfc-editor.org/rfcsearch.html</p>
[RFC3986]	<p>RFC 3986 <i>Uniform Resource Identifier (URI): Generic Syntax</i> by T. Berners-Lee, R. Fielding and L. Masinter Date: January 2005 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: http://www.rfc-editor.org/rfcsearch.html</p>
[RFC3987]	<p>RFC 3987 <i>Internationalized Resource Identifiers (IRIs)</i>, by M. Duerst and M. Suignard Date: January 2005 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: http://www.rfc-editor.org/rfcsearch.html</p>
[RFC5246]	<p>RFC 5246 <i>The Transport Layer Security (TLS) Protocol Version 1.2</i>, by T. Dierks and E. Rescorla Date: August 2008 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: http://www.rfc-editor.org/rfcsearch.html</p>
[RFC6068]	<p>RFC 6068 <i>The mailto URL scheme</i> Date: October 2010 Produced by: Internet Engineering Task Force (IETF), Network Working Group Available at: http://www.rfc-editor.org/rfcsearch.html</p>
[TAPPI T480]	<p>TAPPI T480 <i>Specular Gloss of Paper and Paperboard at 75 Degrees, Test Method T 480 om-15</i> Date: December 2009 Produced by: TAPPI Available at: http://www.tappi.org</p>
[TAPPI T519]	<p>TAPPI T519 <i>Diffuse Opacity of Paper (d/o paper backing), Test Method T 519 om-17</i> Date: December 2009 Produced by: TAPPI Available at: http://www.tappi.org</p>
[TAPPI T527]	<p>TAPPI T527 <i>Color of Paper and Paperboard (d/o, C/2), Test Method T 527 om-02</i> Date: - Produced by: TAPPI Available at: http://www.tappi.org</p>
[TAPPI T560]	<p>TAPPI T560 <i>CIE whiteness and tint of paper and paperboard (d/o geometry, C/2 illuminant/observer), Test Method T 560 om-10</i> Date: December 2009 Produced by: TAPPI Available at: http://www.tappi.org</p>

Table H.1: References (Sheet 8 of 8)

TERM	DEFINITION
[TIFF6]	<p><i>TIFF Revision 6.0</i></p> <p>Date: June 1992 Produced by: Adobe Systems Incorporated Available at: https://www.loc.gov/preservation/digital/formats/fdd/fdd000022.shtml#notes</p>
[vCard]	<p><i>Standard file format for electronic business cards.</i></p> <p>Date: N/A Produced by: Versit Consortium Refer to : https://en.wikipedia.org/wiki/VCard</p>
[XJDF]	<p><i>Exchange Job Definition Format</i> <i>Version 2.0</i></p> <p>Date: January 2018 Produced by: International Cooperation for the Integration of Processes in Prepress, Press and Postpress (CIP4) Available at: http://www.cip4.org</p>
[XML Tutorial]	<p><i>Extensible Markup Language Tutorials and Guides</i></p> <p>Date: - Produced by: XMLFiles Available at: https://www.xmlfiles.com/</p>
[XML]	<p><i>Extensible Markup Language (XML) 1.0 (Fifth Edition)</i> <i>Version (W3C Recommendation of 26 November 2008)</i></p> <p>Date: 26 November 2008 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)</i> <i>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</i> <i>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/xmlschema-0/ http://www.w3.org/TR/xmlschema-1/ http://www.w3.org/TR/xmlschema-2/</p>
[XPath]	<p><i>XML Path Language (XPath) 2.0 (Second Edition)</i> <i>Version W3C Recommendation 14 December 2010</i></p> <p>Date: 14 December 2010 Produced by: World Wide Web Consortium (W3C) Available at: https://www.w3.org/TR/xpath20/</p>
[ZIP]	<p><i>File compression and archiving</i> <i>.ZIP File Format Specification - Version 6.3.4</i></p> <p>Date: 1 October 2014 Produced by: PKWARE Inc. Available at: https://support.pkware.com/display/PKZIP/APPNOTE</p>

CIP4



ORGANIZATION

INTEGRATION THROUGH COOPERATION



HEIDELBERG



Kodak



RICOH



cip4.org