

Simulation Interoperability Standards Organization (SISO)

Standard for: Core Manufacturing Simulation Data — UML Model

SISO-STD-008-2010

20 September 2010

**Prepared by:
Core Manufacturing Simulation Data
Product Development Group**

Copyright © 2010 by the Simulation Interoperability Standards Organization (SISO), Inc.

P.O. Box 781238

Orlando, FL 32878-1238, USA

All rights reserved.

Permission is hereby granted for SISO developing committee participants to reproduce this document for purposes of SISO product development activities only. Prior to submitting this document to another standards development organization for standardization activities, permission must first be obtained from the SISO Standards Activity Committee (SAC). Other entities seeking permission to reproduce this document, in whole or in part, must obtain permission from the SISO Executive Committee (EXCOM).

SISO EXCOM

P.O. Box 781238

Orlando, FL 32878-1238, USA

Revision History

Version	Section	Date	Description
SISO-STD-008-2008	All	2/13/2009	Initial draft
SISO-STD-008-2009	All	5/11/2009	Submit as a part of the balloting package
SISO-STD-008-2010-DRAFT-V1.0	All	4/30/2010	Incorporate modifications generated from ballot comment resolution
SISO-STD-008-2010	All	9/20/2010	Approved standard

TABLE OF CONTENTS

1	INTRODUCTION	10
1.1	PURPOSE	10
1.2	SCOPE	10
1.3	OBJECTIVE	10
1.4	INTENDED AUDIENCE	10
1.5	ACKNOWLEDGEMENTS	11
2	REFERENCES	12
2.1	SISO REFERENCES	12
2.2	OTHER REFERENCES	12
3	DEFINITIONS	13
4	ACRONYMS AND ABBREVIATIONS	15
5	CORE MANUFACTURING SIMULATION DATA (CMSD) INFORMATION MODEL OVERVIEW	16
5.1	CMSD - DEFINED USING UML	16
5.2	CMSD ENTITIES - MANUFACTURING CONCEPTS REALIZED WITH UML CLASS DEFINITIONS	17
5.3	THE CMSD DOCUMENT CONCEPT	17
5.3.1	<i>Unique and Limited Unique Entities</i>	17
5.3.2	<i>Conceptual Representation of a CMSD Document</i>	18
6	THE CORE MANUFACTURING SIMULATION DATA INFORMATION MODEL	20
6.1	BASIC STRUCTURES PACKAGE	21
6.1.1	<i>Diagrams</i>	21
6.1.1.1	Classes Supporting the Definition of Measured Values and their Units	21
6.1.1.2	Classes Supporting Property Attributes Definition	21
6.1.1.3	Classes Supporting Distribution Information Definition	22
6.1.1.4	Classes Supporting Contact Information Definition	22
6.1.1.5	Classes Supporting Precedence Constraint Definition	23
6.1.1.6	Classes Supporting the Definition of Order, Job, and Process Related Information	23
6.1.1.7	Classes Supporting Layout Information Definition	24
6.1.1.8	Classes Supporting the Definition of Information about Reference Materials	24
6.1.1.9	Classes that Define Other Basic Structures	25
6.1.2	<i>Classes</i>	25
6.1.2.1	Address Class	25
6.1.2.2	AreaType Class	26
6.1.2.3	BoundaryDefinition Class	26
6.1.2.4	ColorHighlight Class	27
6.1.2.5	<i>CommunicationMethod</i> Class	28
6.1.2.6	ContactInformation Class	28
6.1.2.7	ContactParty Class	28
6.1.2.8	Coordinate2D Class	29
6.1.2.9	Coordinate3D Class	29
6.1.2.10	CostAllocationData Class	30
6.1.2.11	CurrencyType Class	30
6.1.2.12	Distribution Class	31
6.1.2.13	DistributionDefinition Class	31
6.1.2.14	DistributionParameter Class	32
6.1.2.15	Duration Class	32
6.1.2.16	ElapsedTimeType Class	33
6.1.2.17	Email Class	34
6.1.2.18	Event Class	34

6.1.2.19	GrossDimensions Class	35
6.1.2.20	ImageResolution Class	35
6.1.2.21	ItemPackaging Class	36
6.1.2.22	JobConstraint Class	36
6.1.2.23	LengthType Class	37
6.1.2.24	LocationDefinition Class	37
6.1.2.25	LotInformation Class	38
6.1.2.26	MachineProgramData Class	38
6.1.2.27	MaintenanceProcessConstraint Class	39
6.1.2.28	PartGroup Class	39
6.1.2.29	PerMeasuredAmountPackaging Class	40
6.1.2.30	PerPiecePackaging Class	41
6.1.2.31	Phone Class	41
6.1.2.32	PowerType Class	41
6.1.2.33	<i>PrecedenceConstraint</i> Class	42
6.1.2.34	ProcessConstraint Class	43
6.1.2.35	Property Class	43
6.1.2.36	QuantityType Class	44
6.1.2.37	ReferenceMaterial Class	44
6.1.2.38	<i>ReferenceProperty</i> Class	45
6.1.2.39	RequiredApplication Class	46
6.1.2.40	ResourcesRequired Class	46
6.1.2.41	ShapeLabelDefinition Class	47
6.1.2.42	<i>SimpleProperty</i> Class	48
6.1.2.43	SpatialDimension Class	48
6.1.2.44	SpeedType Class	49
6.1.2.45	<i>StochasticProperty</i> Class	49
6.1.2.46	TemperatureType Class	50
6.1.2.47	VariableCostData Class	50
6.1.2.48	VolumeType Class	51
6.1.2.49	WeightType Class	51
6.2	BASIC TYPES PACKAGE	52
6.2.1	<i>Diagrams</i>	52
6.2.1.1	Extended Primitive Data Types	52
6.2.1.2	Types Related to Values for Units of Measure	53
6.2.1.3	Types Supporting the Definition of Metadata Information	54
6.2.1.4	Types Supporting Layout Information Definition	55
6.2.1.5	Other Basic Data Types	56
6.2.2	<i>Classes</i>	56
6.2.2.1	AreaUnit Class	56
6.2.2.2	BaseLocation Class	56
6.2.2.3	BasicShapeType Class	57
6.2.2.4	Boolean Class	57
6.2.2.5	CardinalitySpecification Class	57
6.2.2.6	ColorDefinition Class	57
6.2.2.7	ColorHexString Class	57
6.2.2.8	ColorName Class	57
6.2.2.9	ConnectionType Class	57
6.2.2.10	CoordinateSystem Class	58
6.2.2.11	CostCategory Class	58
6.2.2.12	CostType Class	58
6.2.2.13	CurrencyUnit Class	58
6.2.2.14	Date Class	58
6.2.2.15	Day Class	59
6.2.2.16	Decimal Class	59
6.2.2.17	GraphicDescriptionType Class	59

6.2.2.18	Identifier Class	59
6.2.2.19	Integer Class	59
6.2.2.20	InventoryItemType Class	59
6.2.2.21	JobStatus Class	59
6.2.2.22	LengthUnit Class	59
6.2.2.23	LayoutLengthUnit Class	60
6.2.2.24	NonNegativeInteger Class	60
6.2.2.25	OrderStatus Class	60
6.2.2.26	PartProductionStatus Class	60
6.2.2.27	PowerUnit Class	60
6.2.2.28	PrecedenceRelationship Class	60
6.2.2.29	ProcessGroupType Class	61
6.2.2.30	PropertyExtensibleEntity Class	61
6.2.2.31	PropertyType Class	61
6.2.2.32	ReferenceTypeName Class	61
6.2.2.33	ResourceStatus Class	61
6.2.2.34	ResourceType Class	62
6.2.2.35	SegmentType Class	62
6.2.2.36	ShapeDescriptionType Class	62
6.2.2.37	SimpleDataTypeName Class	62
6.2.2.38	SpeedUnit Class	62
6.2.2.39	String Class	62
6.2.2.40	TemperatureUnit Class	62
6.2.2.41	TextAnchorLocation Class	63
6.2.2.42	Time Class	63
6.2.2.43	Timestamp Class	63
6.2.2.44	TimeUnit Class	63
6.2.2.45	UnitTypeName Class	63
6.2.2.46	UnlimitedCardinality Class	63
6.2.2.47	URI Class	64
6.2.2.48	VolumeUnit Class	64
6.2.2.49	WeightUnit Class	64
6.3	CMSD PACKAGE	65
6.3.1	<i>Diagrams</i>	65
6.3.1.1	Top Level CMSD Packages	65
6.3.2	<i>Classes</i>	66
6.4	CONCEPTUAL FRAMEWORK PACKAGE	67
6.4.1	<i>Diagrams</i>	67
6.4.1.1	Conceptual Framework Classes	67
6.4.2	<i>Classes</i>	68
6.4.2.1	<i>Entity</i> Class	68
6.4.2.2	<i>IdentifiableEntity</i> Class	68
6.4.2.3	<i>LimitedUniqueEntity</i> Class	69
6.4.2.4	<i>ReferencingEntity</i> Class	70
6.4.2.5	<i>UniqueEntity</i> Class	70
6.5	DOCUMENT DEFINITION PACKAGE	71
6.5.1	<i>Diagrams</i>	71
6.5.1.1	Classes of the Document Definition Package	71
6.5.2	<i>Classes</i>	72
6.5.2.1	CMSDDocument Class	72
6.5.2.2	CMSDDocumentReference Class	72
6.5.2.3	DataSection Class	73
6.5.2.4	HeaderSection Class	74
6.5.2.5	Metadata Class	75
6.5.2.6	UnitDefaults Class	75

6.6	ENTITY REFERENCE DEFINITION PACKAGE	77
6.6.1	<i>Diagrams</i>	78
6.6.1.1	The AbstractEntityReference Class and the Basic Entity Reference Classes	78
6.6.1.2	Complex Entity Reference Classes	80
6.6.2	<i>Classes</i>	81
6.6.2.1	AbstractEntityReference Class	81
6.6.2.2	Basic Entity Reference Classes - BillOfMaterialsReference, DistributionDefinitionReference, InventoryItemClassReference, InventoryItemReference, JobReference, LayoutElementReference, MaintenancePlanReference, PartReference, PartTypeReference, ProcessPlanReference, PropertyDescriptionReference, ReferenceMaterialReference, ResourceClassReference, ResourceReference, SetupChangeoverReference, and SetupDefinitionReference	81
6.6.2.3	BillOfMaterialsComponentReference Class	82
6.6.2.4	CalendarReference Class	83
6.6.2.5	MaintenanceProcessReference Class	83
6.6.2.6	OrderInformationReference Class	84
6.6.2.7	ProcessReference Class	85
6.6.2.8	ScheduleInformationReference Class	85
6.6.2.9	SkillReference Class	86
6.7	LAYOUT PACKAGE	87
6.7.1	<i>Diagrams</i>	87
6.7.1.1	LayoutElement and Related Classes	87
6.7.1.2	ShapeDescription and Related Classes	88
6.7.1.3	TransformationList and Related Classes	89
6.7.1.4	GraphicDescription and Related Classes	90
6.7.1.5	The TextualAnnotation Class	90
6.7.1.6	BasicShape and Related Classes	91
6.7.1.7	SegmentShape and Related Classes	92
6.7.2	<i>Classes</i>	93
6.7.2.1	BasicShape Class	93
6.7.2.2	Box Class	93
6.7.2.3	Circle Class	94
6.7.2.4	CurvedSegment Class	94
6.7.2.5	GraphicDescription Class	95
6.7.2.6	ImageGraphic Class	96
6.7.2.7	Layout Class	96
6.7.2.8	LayoutElement Class	97
6.7.2.9	LayoutObject Class	98
6.7.2.10	ModelGraphic Class	98
6.7.2.11	Placement Class	99
6.7.2.12	Polygon Class	99
6.7.2.13	Rotation Class	100
6.7.2.14	Scaling Class	100
6.7.2.15	SegmentShape Class	101
6.7.2.16	ShapeDescription Class	102
6.7.2.17	StraightSegment Class	103
6.7.2.18	TextualAnnotation Class	104
6.7.2.19	TransformationList Class	105
6.7.2.20	TransformationOperation Class	105
6.7.2.21	Translation Class	106
6.8	METADATA PACKAGE	107
6.8.1	<i>Diagrams</i>	107
6.8.1.1	PropertyDescription and Related Classes	107
6.8.2	<i>Classes</i>	108
6.8.2.1	PropertyCardinality Class	108

6.8.2.2	PropertyDataTypeDescription Class.....	108
6.8.2.3	PropertyDescription Class.....	109
6.9	PART INFORMATION PACKAGE	111
6.9.1	<i>Diagrams</i>	111
6.9.1.1	The Part Class and PartType Class.....	111
6.9.1.2	BillOfMaterials and Related Classes.....	112
6.9.1.3	InventoryItem and Related Classes	113
6.9.2	<i>Classes</i>	113
6.9.2.1	BillOfMaterials Class	113
6.9.2.2	BillOfMaterialsComponent Class.....	114
6.9.2.3	InventoryItem Class.....	115
6.9.2.4	InventoryItemClass Class.....	116
6.9.2.5	Part Class	117
6.9.2.6	PartType Class	118
6.10	PRODUCTION OPERATIONS PACKAGE	119
6.10.1	<i>Diagrams</i>	119
6.10.1.1	Order and Job Related Classes	119
6.10.1.2	The Schedule Class and Related Classes.....	121
6.10.2	<i>Classes</i>	122
6.10.2.1	Job Class	122
6.10.2.2	JobEffortDescription Class.....	123
6.10.2.3	Order Class	124
6.10.2.4	OrderLine Class.....	125
6.10.2.5	OrderLinePartDescription Class.....	126
6.10.2.6	OrderLineServiceDescription Class	127
6.10.2.7	Schedule Class	127
6.10.2.8	ScheduleItem Class	128
6.10.2.9	ScheduleItemEffortDescription Class	130
6.11	PRODUCTION PLANNING PACKAGE	131
6.11.1	<i>Diagrams</i>	131
6.11.1.1	Calendar and Shift Related Classes.....	131
6.11.1.2	ProcessPlan, MaintenancePlan, and Related Classes.....	132
6.11.2	<i>Classes</i>	133
6.11.2.1	AvailabilityException Class.....	133
6.11.2.2	Break Class	133
6.11.2.3	Calendar Class	134
6.11.2.4	Holiday Class	135
6.11.2.5	MaintenancePlan Class.....	135
6.11.2.6	MaintenanceProcess Class.....	136
6.11.2.7	MaintenanceProcessGroup Class.....	137
6.11.2.8	Process Class	138
6.11.2.9	ProcessGroup Class	139
6.11.2.10	ProcessPlan Class	140
6.11.2.11	Shift Class.....	141
6.11.2.12	ShiftSchedule Class	142
6.12	RESOURCE INFORMATION PACKAGE	143
6.12.1	<i>Diagrams</i>	143
6.12.1.1	The Resource Class and ResourceClass Class	143
6.12.1.2	ResourceGroupInformation and Related Classes	144
6.12.1.3	The SkillDefinition Class and SkillLevel Class	144
6.12.1.4	SetupDefinition and Related Classes	145
6.12.2	<i>Classes</i>	145
6.12.2.1	Connection Class.....	145
6.12.2.2	NewSetup Class	146
6.12.2.3	Resource Class.....	147

6.12.2.4	ResourceClass Class.....	148
6.12.2.5	ResourceGroupInformation Class.....	149
6.12.2.6	ResourceGroupMember Class.....	149
6.12.2.7	SetupChangeoverDefinition Class.....	150
6.12.2.8	SetupDefinition Class.....	150
6.12.2.9	SkillDefinition Class.....	151
6.12.2.10	SkillLevel Class.....	151
6.13	SUPPORT PACKAGE	153
6.13.1	<i>Diagrams</i>	153
6.13.1.1	Support Package Sub-packages.....	153
6.13.2	<i>Classes</i>	154
ANNEX A	BIBLIOGRAPHY	155
ANNEX B	CLASS CROSS REFERENCE	156
ANNEX C	UML QUICK REFERENCE.....	165

1 Introduction

This product, Core Manufacturing Simulation Data (CMSD), addresses interoperability between simulation systems and other manufacturing applications. The CMSD information model is a standard representation for core manufacturing simulation data. It provides neutral structures for the efficient exchange of manufacturing data in a simulation environment. These neutral structures can be used to support the integration of simulation software with other manufacturing applications.

The specification of the CMSD information model is presented using two different methods: (1) the information model defined using the Unified Modeling Language (UML); and (2) the information model defined using the XML Schema definition language. The information model defined using UML is presented in this document.

The eXtensible Markup Language (XML), a specification supported by the World Wide Web Consortium (W3C), is a way to represent textual information in a document using an extensible set of markup tags that specify enhanced semantic content for the information in the document. The XML Schema definition language, which is also a specification supported by the W3C, provides a means to define a specific kind of document by defining a specific set of markup tags and the allowable organization of information using those tags. The XML Schema representation of the CMSD information model is not the subject of this document.

1.1 Purpose

The purposes of this product include:

- enabling data exchange between simulation applications and other software applications;
- supporting the construction of manufacturing simulators;
- supporting the testing and evaluation of manufacturing software; and
- enabling greater manufacturing software application interoperability.

1.2 Scope

The CMSD information model describes the essential entities in the manufacturing domain and the relationships between those entities needed to create manufacturing-oriented simulations. This model facilitates the exchange of information between manufacturing-oriented simulations and other applications in manufacturing domains such as process planning, scheduling, inventory management, production management, and plant layout. The model is not intended to be an all-inclusive definition of either the entire manufacturing domain or simulation domain.

1.3 Objective

The primary objective of this standard is to provide a data specification that enables the efficient exchange of manufacturing life-cycle data in a simulation environment. The objective is intended to:

- foster the development and use of simulations in manufacturing operations;
- facilitate data exchange between simulation and other manufacturing software applications;
- enable and facilitate better testing and evaluation of manufacturing software; and
- increase manufacturing application interoperability.

1.4 Intended Audience

The primary audience for this document is the manufacturing-oriented modeling & simulation community. Other communities of interest, although not the intended primary audience, are encouraged to leverage the standard described here for use in their domains.

1.5 Acknowledgements

This document was created as a community effort by the Core Manufacturing Simulation Data (CMSD) Product Development Group (PDG). This PDG was chartered by the Simulation Interoperability Standards Organization (SISO) Standards Activity Committee (SAC) in September 2004. This document would not have been possible without the hard work and dedicated efforts of the following individuals:

Product Development Group Officers

Swee Leong (Chair)

Frank Riddick (Vice Chair)

Y. Tina Lee (Secretary)

Peggy Gravitz (SAC TAD)

Jane Bachman (Acting TAD)

Drafting Group Team

Frank H. Riddick (Co-Editor)

Y. Tina Lee (Co-Editor)

Balloting Group

Jane Bachman	Nils Bengtsson	Damien Bertot
Bob Brown	Michael Burnette	Terry Christian
James Coolahan	David Cutts	Steve Deflitch
Em delaHostria	Jon Fournier	Peggy Gravitz
Per Gullander	Charley Harrell	Juhani Heilala
Jim Hollenbach	Sanjay Jain	Björn Johansson
Deo Kibira	Yung-Tsun Tina Lee	Mike Lightner
Jim Lorenz	Roberto Lu	Gregory Martin
James McCall	Chuck McLean	Bjorn Moller
Lihong Qiao	Guillaume Radde	Frank Riddick
Tom Scotton	Jay Seddon	Daniel Semere
Rick Severinghaus	Anders Skoogh	Young-Jun Son
Eugene Stoudenmire	Steffen Strassburger	Simon Taylor
William Tucker	Bill Waite	Ralph Weber

2 References

2.1 SISO References

	Document Number	Title
1	SISO-ADM-002-2008.1	SISO Policies & Procedures
2	SISO-ADM-003-2008	SISO Balloted Products Development and Support Process (BPDSP)
3	SISO-ADM-005-2008	The Style and Format of SISO Documents
4	SISO-PDG-PN-CMSD_2004-008-05	CMSD Product Nomination

2.2 Other References

	Document Number	Title
1	ANSI 260.1- 1993	American National Standard Letter Symbols for Units of Measurement (SI Units, Customary Inch-Pound Units, and Certain Other Units)
2	IEEE Std 280 -1985	IEEE Standard Letter Symbols for Quantities Used in Electrical Science and Electrical Engineering
3	IETF Internet Engineering Task Force, RFC 3986	Uniform Resource Identifier (URI) : Generic Syntax
4	ISO 31:0 - 1992 (E) – ISO 31:13 - 1992	Quantities and Units
5	ISO 1000 - 1992	SI Units and Recommendations for the Use of Their Multiples and of Certain Other Units
6	ISO 3166 Part 1 - 2006/Cor 1 - 2007 Part 2 - 2007 Part 3 - 1999	Codes for the Representation of Names of Countries and Their Subdivisions -- Part 1: Country Codes Part 2: Country Subdivision Code Part 3: Code for Formerly Used Names of Countries
7	ISO 4217 - 2008	Codes for the Representation of Currencies and Funds
8	ISO 8601 - 2004	Data Elements and Interchange Formats – Information Interchange – Representation of Dates and Times
9	NIST Special Publication 330, 2008	The International System of Units (SI)

	Edition	
10	NIST Special Publication 811, 2008	Guide for the Use of the International System of Units (SI)
11	Object Management Group (OMG) UML 2.1.2, 2007	Unified Modeling Language (UML)
12	World Wide Web Consortium (W3C) CSS2	Cascading Style Sheets, Level 2, CSS2 Specification
13	World Wide Web Consortium (W3C) HTML 4.01	HyperText Markup Language (HTML) Specification
14	World Wide Web Consortium (W3C) XML Schema Datatypes	XML Schema Part 2: Datatypes Second Edition

3 Definitions

Bill of materials	A description of the hierarchical relationships between a part and its subcomponents.
Calendar	A long term focused collection of shift and holiday information that, taken together, specify the time periods during which production is and is not expected to take place.
CMSD document	An aggregation of information that is organized based on the CMSD specification and that is suitable for exchange or archival.
Cost	The expense incurred by an enterprise due to its performing some manufacturing activity.
Distribution	The name and parameter values that define a statistical distribution. A statistical distribution is a mathematical function where: 1) the range of possible values of the function is known, and; 2) the probability that a random input to the domain of the function will produce an output value in a subset of the range is also known.
Entity	An abstraction used in the CMSD information model to represent a common manufacturing or business concept.
Event	The documentation of the planned or actual occurrence of some phenomenon or the reaching of some milestone.
Identifier	An attribute of an entity that is used to differentiate that entity from other entities of the same kind.
Inventory item	A part or (non-employee) resource for which information about its availability for production activities is tracked.
Inventory item type	Information about a kind of part or kind of (non-employee) resource that can be an inventory item.

Job	A request for production-related activities to take place, originating from a person or organization internal to the manufacturing enterprise.
Layout	A representation of the spatially-relevant characteristics of, and relationships between, the manufacturing resources that are a part of a manufacturing facility.
Limited unique entity	An entity whose instances are unique but only within the scope of its parent entity.
Lot	Information about a group of parts that were manufactured together or obtained together, and that have some important characteristic.
Machine program	A set of instructions that allow a computer-controlled machine tool to perform a specific manufacturing function.
Maintenance plan	A collection of maintenance processes that provide the necessary instructions for maintaining a (non-employee) manufacturing resource.
Maintenance process	A manufacturing activity or group of manufacturing activities that perform a corrective or preventive maintenance operation on a resource.
Metadata	Information about the format and value space that is allowable for a given property attribute.
Order	A request for products or services originating from a person or organization external to the manufacturing enterprise.
Parent entity	An entity that has other entities nested within it. A parent entity defines a scope for determining the uniqueness of multiple instances of the same kind of entity that may be nested within it.
Part	A raw material or sub-component used in or produced by some stage of production, or an end product that is the final objective of production.
Part type	Information about the characteristics of a specific kind of part.
Process	A manufacturing activity or group of manufacturing activities that either: 1) transforms a part from a known state/condition to another known state/condition; 2) transports a part from a known location to another location; 3) verifies that a part is in a known state state/condition or location, or; 4) all of the above.
Process Plan	A collection of processes that provide the necessary instructions for producing a part.
Property	A means for extending the information that can be associated with an entity by allowing name and value information for a noteworthy characteristic of the entity to be associated with that entity.
Property attribute	A characteristic of an entity that was specified using property information.
Resource	A piece of equipment or an employee that is performing or is to perform a manufacturing activity.
Resource class	Information about the characteristics of a specific kind of resource.
Schedule	A plan containing a time-ordered collection of production activities, and/or the results obtained by carrying out such a plan.
Shift	A time period during a day of the week when production activities are to take place and a specification of the days on which this time period is applicable.
Shift schedule	An intermediate term collection of shift and holiday information specifying when production is and is not expected to take place.
Unique entity	An entity whose instances are unique within the scope of a CMSD document.

4 Acronyms and Abbreviations

ADM	Administrative procedures
ANSI	American National Standards Institute
BPDSP	Balloted Products Development and Support Process
CC	Conference Committee
CMSD	Core Manufacturing Simulation Data
CSS2	Cascading Styling Sheet, Level 2
DG	Drafting Group
EXCOM	Executive Committee
FF	Finish-To-Finish precedence relationship
FS	Finish-To-Start precedence relationship
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
ISO	International Organization for Standardization
M&S	Modeling and Simulation
NIST	National Institute of Standards and Technology
PDG	Product Development Group
PN	Product Nomination
RGB	Red, Green, & Blue light intensity values that define a specific color
RFC	Request for Comments
SF	Start-To-Finish precedence relationship
SI	International System of Units
SISO	Simulation Interoperability Standards Organization
SS	Start-To-Start precedence relationship
STD	Standard
TAD	Technical Area Director
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
W3C	World Wide Web Consortium
XML	eXtensible Markup Language

5 Core Manufacturing Simulation Data (CMSD) Information Model Overview

5.1 CMSD - Defined Using UML

The Unified Modeling Language (UML), an industry accepted standard developed and published by the Object Management Group (OMG), is used as the modeling language for the CMSD model. UML is a visually oriented language that can be used to model both hardware and software systems, at different levels of abstraction, and with different levels of detail. Through the use of its thirteen different types of diagrams, UML can be used to model many different aspects of both the structure and behavior of a system. Unlike some other modeling approaches, UML does not define or designate the use of a specific modeling methodology that must be used along with its visual modeling components.

Annex C contains information explaining how to interpret the specific UML diagrams and symbols used in this document to define the CMSD model.

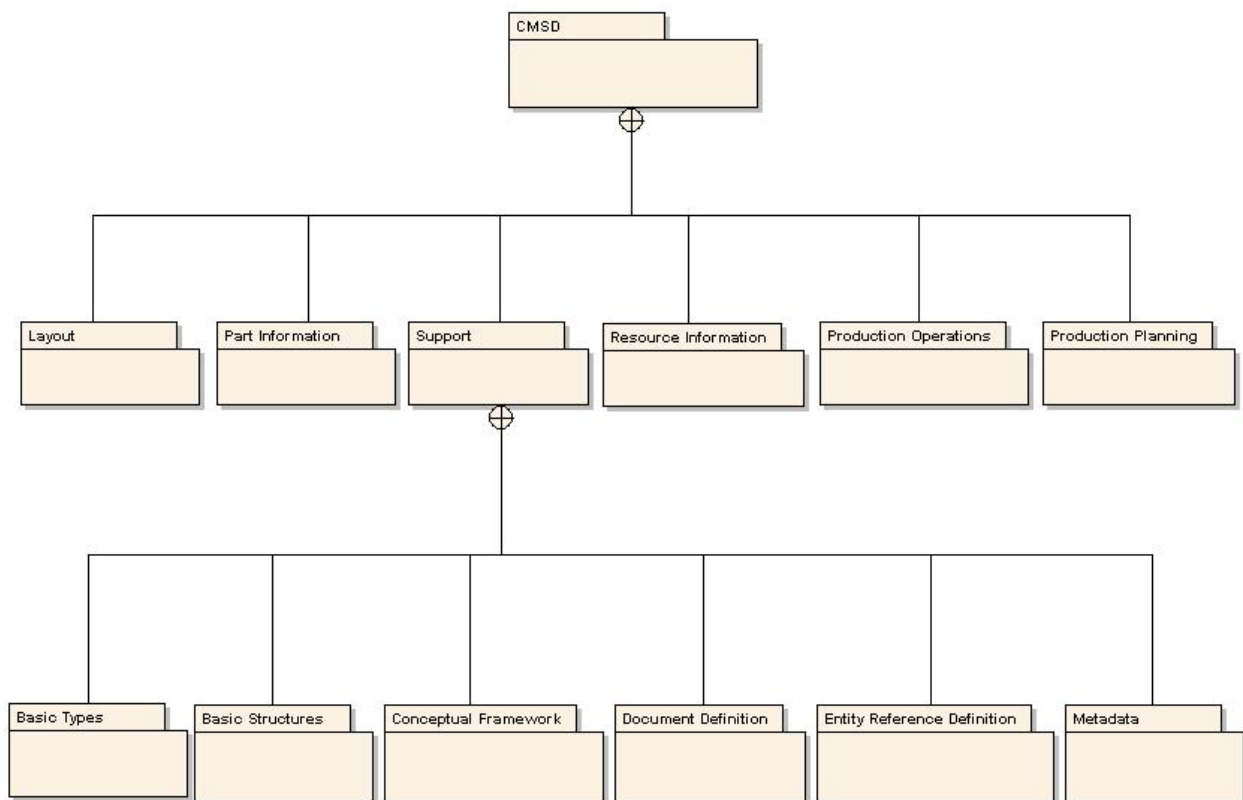


Figure 1: The Packages of the CMSD Model

Figure 1 shows all of the packages that make up the CMSD model. The CMSD model is designed as a suite of interrelated collections of information modeled as UML classes contained within UML packages, presented visually as a series of UML class and package diagrams. The primary function of the UML packages in the CMSD model is to partition and group, by major areas of manufacturing, the class definitions that realize related manufacturing concepts. When necessary to improve model clarity, some packages may be further partitioned into subpackages.

The manufacturing concepts within each package are modeled as UML classes and the characteristics associated with each entity are modeled as UML class attributes. Operation specifications are not used in defining classes and the visibility of all class attributes is public. Each attribute shall have its name and type specified. The multiplicity of each attribute shall be specified if it is other than 1 to 1.

Attributes associated with a class may be defined directly in the class, or may be defined through an aggregation association with another class defined in the model. Abstract classes are used to define higher-level concepts, and concrete classes may be derived from the abstract classes or other concrete classes through generalization (inheritance) relationships. In such cases, the child class “inherits” all attributes defined in the parent class.

UML constraints are applied to classes and/or associations to add restrictions to the definitions of the associated elements. Also, UML notes are used to highlight features of a diagram that need to be taken into account to properly interpret the model.

Within each package, a series of UML class diagrams is used to present details about the content of each class and its relationship to the other classes in the model. A class may appear on more than one diagram, but it is defined only once in its owning package.

In section 6, the definition of each package and its contents are presented. For each package, a series of diagrams is presented depicting the classes and/or subpackages and their interrelationships, followed by descriptions of contents of each class.

5.2 CMSD Entities - Manufacturing Concepts Realized with UML Class Definitions

Many manufacturing concepts are represented in CMSD such as the concept of a part, resource, process, order, job, etc. In CMSD, these concepts are referred to as CMSD entities or just entities. A CMSD entity might be simple and represent a basic fact about some aspect of manufacturing, or it might be complex and represent a composition of more basic entities and other information.

UML classes are used to realize/implement/define the meaning of the concepts represented by a CMSD entity for the CMSD model. Classes are always named using UpperCamelCase. For example, the entity “part” is an end product or component used to produce an end product, and the UML class “Part” realizes that concept in the CMSD Model.

In the package and class definitions of the CMSD model, the descriptions of the concepts are sometimes presented in terms of entity names (for example, process plan) and sometimes in terms of the class name (for example, ProcessPlan.) The interpretation of the CMSD model’s class and package definitions is not dependent on whether the description is presented in terms of entities or the classes that realize those entities.

5.3 The CMSD Document Concept

One of the primary goals of CMSD is to facilitate the definition of collections of structured pieces of information in a way such that these collections are suitable for archival or exchange. The structured pieces of information are referred to (conceptually) as CMSD entities, and the collections of CMSD entities are referred to as CMSD documents.

A CMSD document is a collection of CMSD entities and related information, assembled in a way such that the assemblage is appropriate for archiving or exchange. It also defines the scope for determining the uniqueness of the entities it contains. In a CMSD document, different kinds of information are grouped together in separate areas called “sections”. The header section contains identifying information about a document and its purpose, metadata describing how some CMSD entities may be extended with additional properties, and other information. The data section contains the definition for all of the unique entities that are in a document. Unique entities in CMSD are used to represent major manufacturing concepts like orders, resources, bill of materials, or process plans.

5.3.1 Unique and Limited Unique Entities

All entities defined within the scope of a CMSD document do not have the same level of uniqueness. An entity that is designated as “unique” shall have identifying information that is different from any other entity of the same type within the same document. Unique entities are the only entities that may be directly nested within the data section of a document. In contrast, “limited unique” entities appear nested within an instance

of a unique entity, referred to as the “parent” entity. They have identifying information that shall be different from any other entity of the same type that exists nested within the same parent entity.

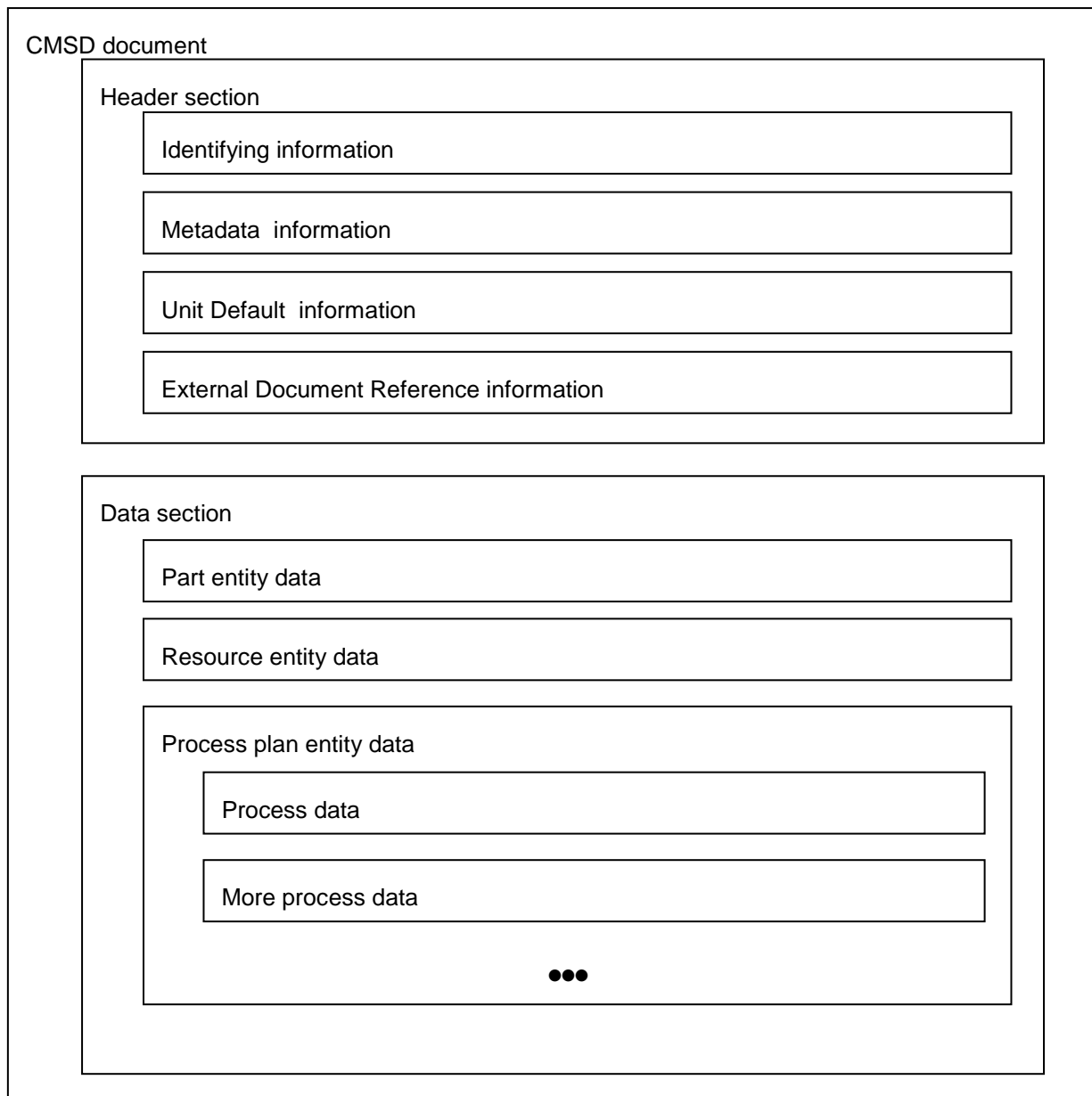
Most of the entities in CMSD are not designated as unique or limited unique, and as such, neither uniqueness designation is applicable to them.

5.3.2 Conceptual Representation of a CMSD Document

Figure 2 presents a conceptual representation of a CMSD document. It shows a CMSD document with a header section and data section nested within it. It also shows that the data section may contain multiple instances of unique CMSD entities. In this example, the unique entities part, resource, and process plan are presented. Some unique entities may contain multiple instances of limited unique entities. In Figure 2, this is depicted by having several process entities nested with the process plan entity.

This diagram is only intended to provide a rough view of the format and structure of an aggregation of information organized as CMSD document. In the different subsections of section 6, the classes that realize the concepts presented here are defined. Specifically in section 6.5, the definition of the CMSDDocument class, which realizes the CMSD document entity concept, is presented, along with diagrams showing its relationships with other CMSD classes.

There are an almost infinite number of possibilities for creating CMSD documents, containing different assemblages of information to support different exchange and archival purposes. The diagrams and class and package descriptions presented in section 6 define the allowable content and organization for CMSD documents and the information contained within them.

**Figure 2: Example of a CMSD Document**

6 The Core Manufacturing Simulation Data Information Model

In the following sections the packages, classes, and relationships that define the CMSD information model are presented. Each package will be presented in a separate section, and the sections are arranged alphabetically. In each of these sections, diagrams depicting the class, subpackage, and relationship definitions are presented. After the diagrams, a description of each class and its attributes, relationships, and constraints is presented.

6.1 Basic Structures Package

The Basic Structures package contains classes and relationships defining simple structures intended to support the definition of more complex structures in other CMSD packages.

6.1.1 Diagrams

6.1.1.1 Classes Supporting the Definition of Measured Values and their Units

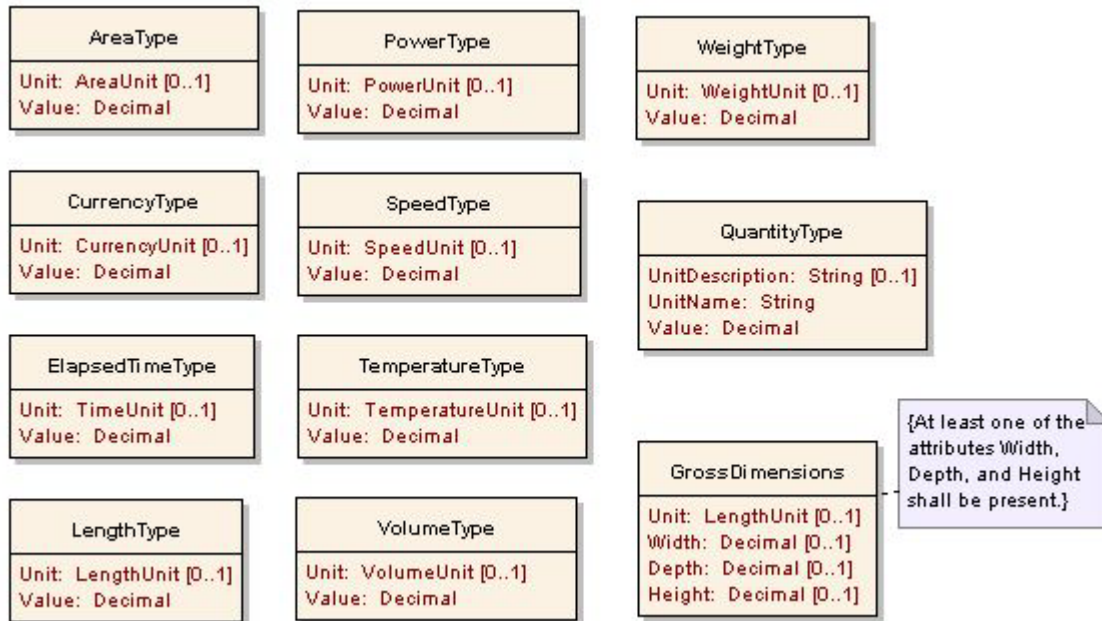


Figure 3: Classes for Measured Values and their Units

6.1.1.2 Classes Supporting Property Attributes Definition

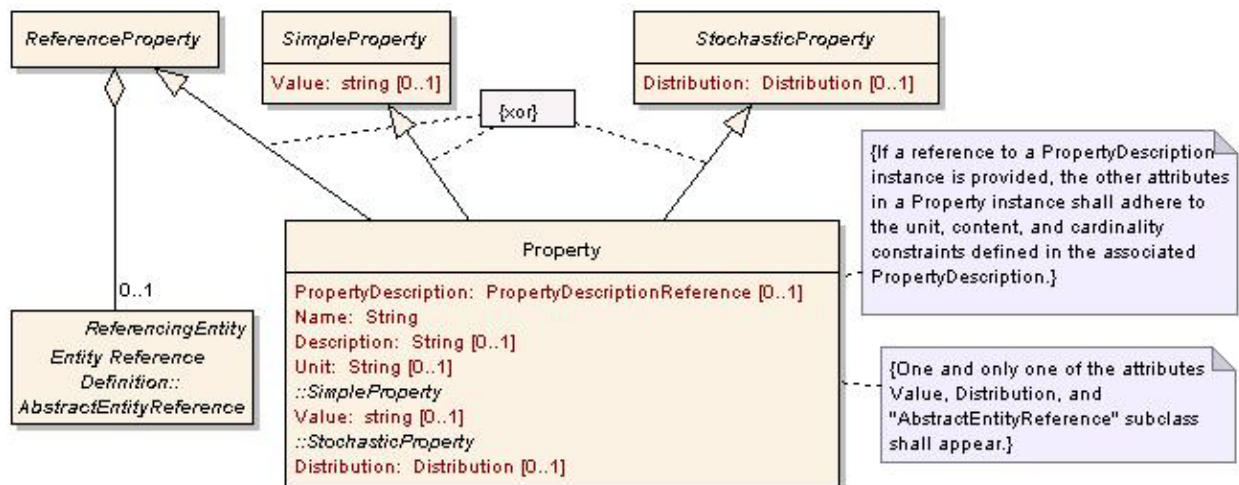


Figure 4: Classes Supporting Property Attribute Definition

6.1.1.3 Classes Supporting Distribution Information Definition

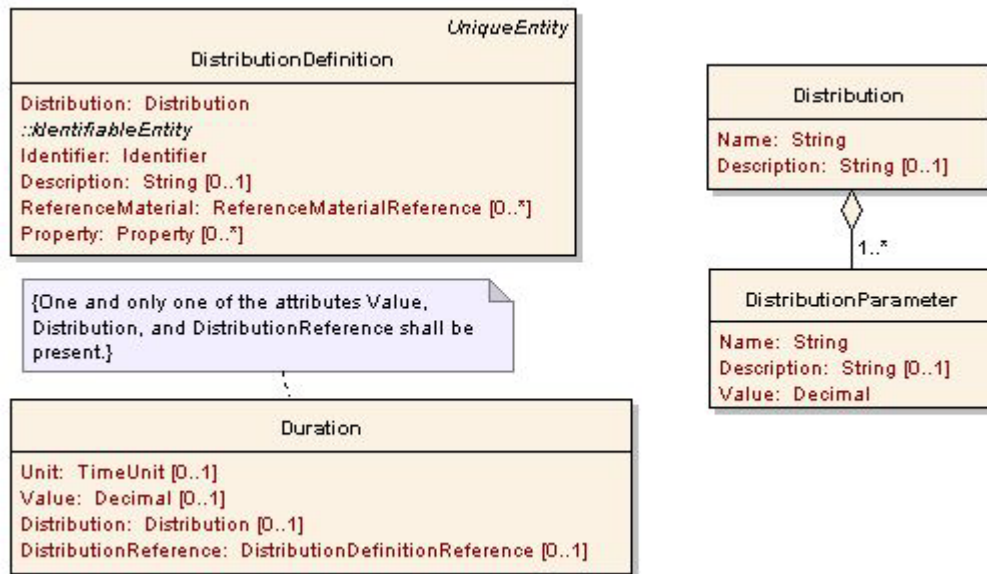


Figure 5: Classes Supporting Distribution Information Definition

6.1.1.4 Classes Supporting Contact Information Definition

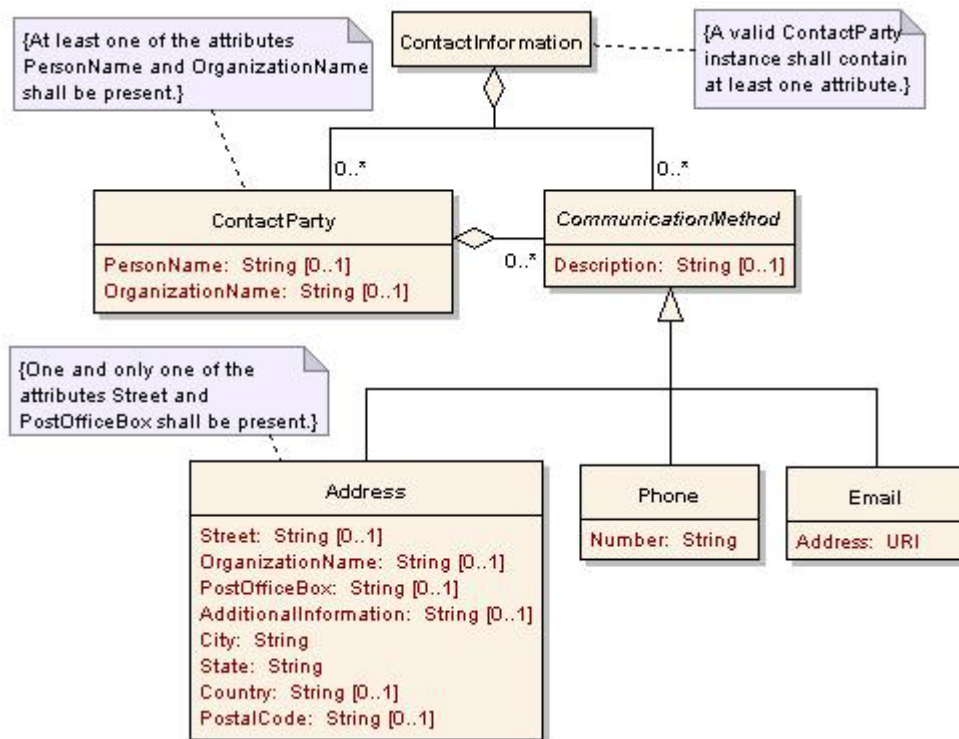


Figure 6: Classes Supporting Contact Information Definition

6.1.1.5 Classes Supporting Precedence Constraint Definition

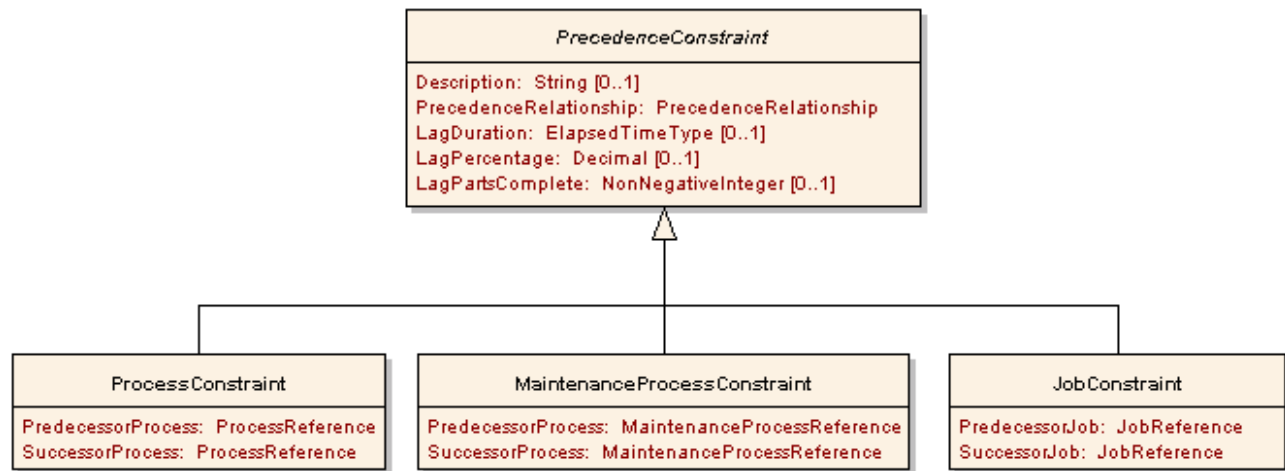


Figure 7: Classes Supporting Precedence Constraint Definition

6.1.1.6 Classes Supporting the Definition of Order, Job, and Process Related Information

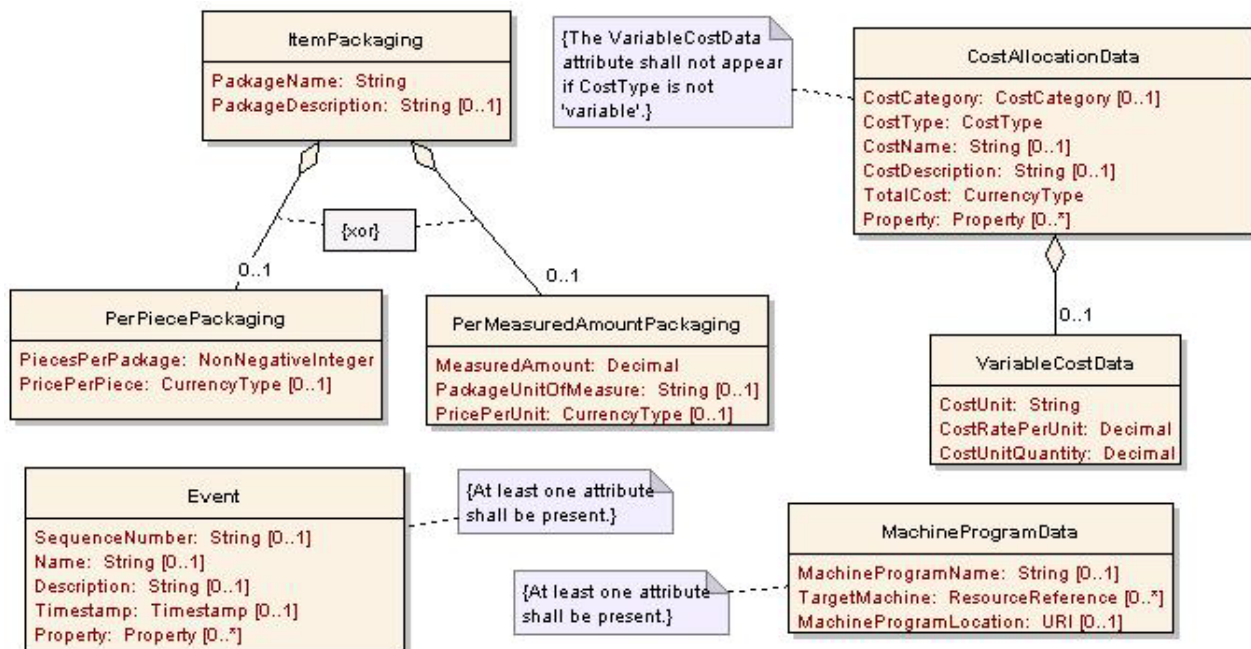


Figure 8: Classes Supporting the Definition of Order, Job, and Process Related Information

6.1.1.7 Classes Supporting Layout Information Definition

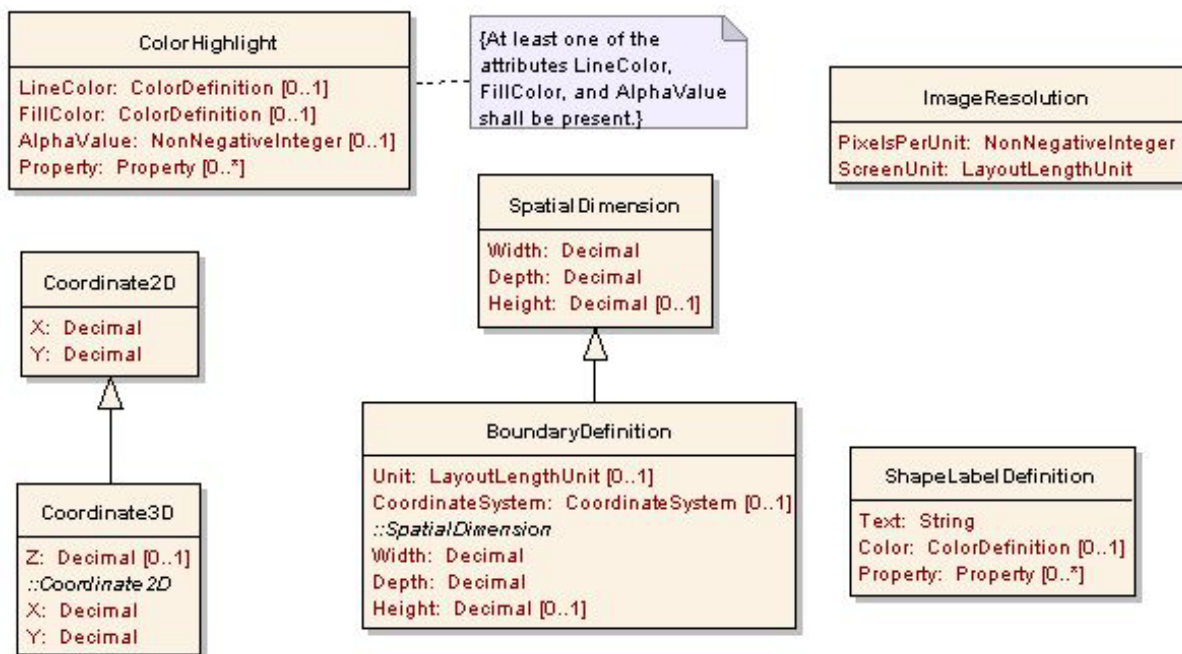


Figure 9: Classes Supporting Layout Information Definition

6.1.1.8 Classes Supporting the Definition of Information about Reference Materials

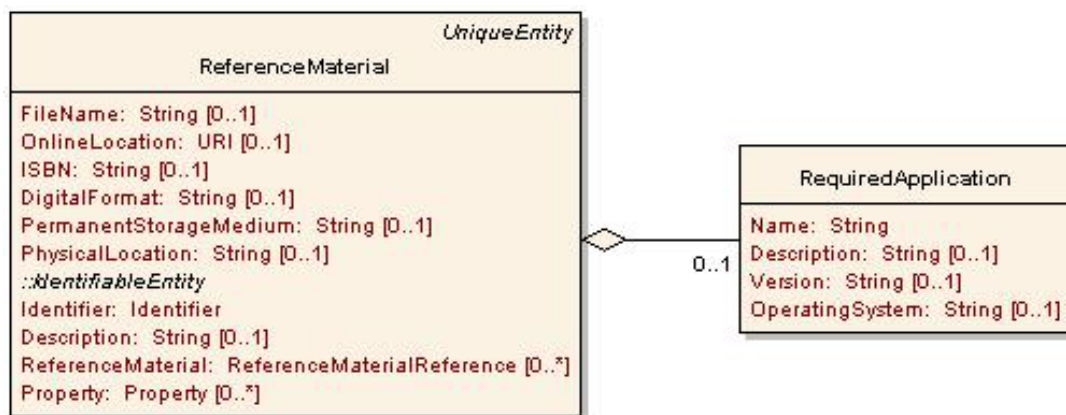


Figure 10: Classes Supporting the Definition of Information about Reference Materials

6.1.1.9 Classes that Define Other Basic Structures

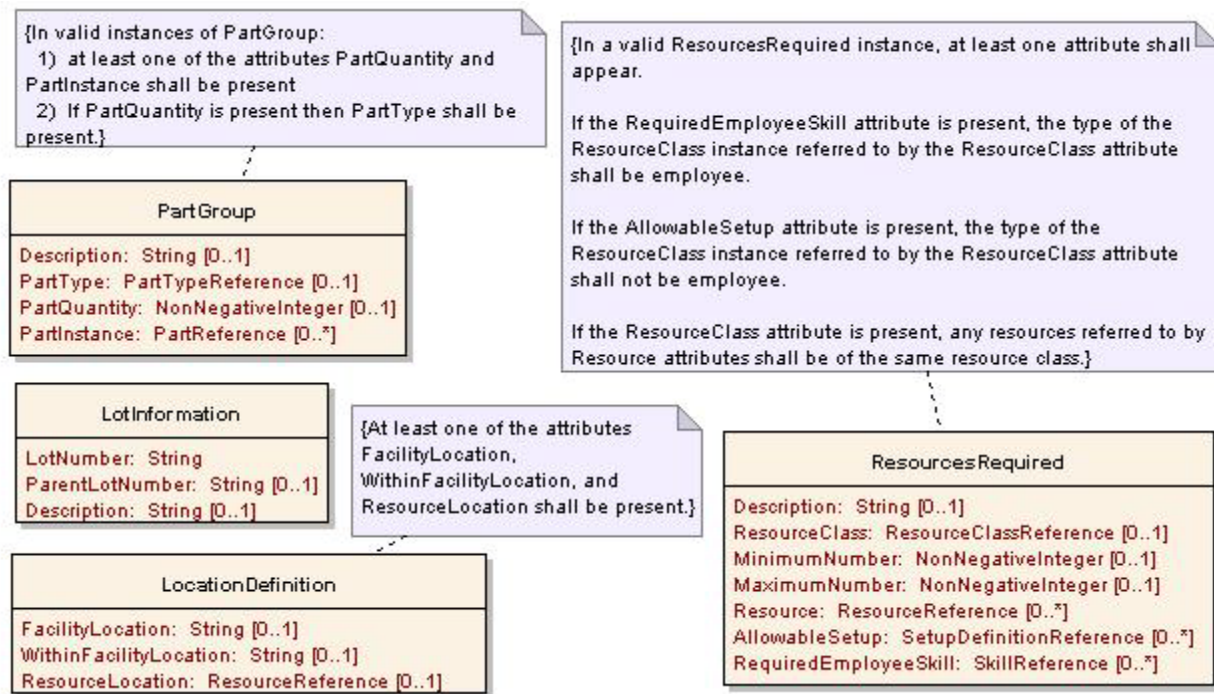


Figure 11: Classes that Define Other Basic Structures

6.1.2 Classes

6.1.2.1 Address Class

The Address class provides a means to specify a street address or post office box that can be used to communicate with a person or organization.

6.1.2.1.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
OrganizationName	String	0 to 1	The name of an organization associated with this address.
Street	String	0 to 1	The street name and number of the address.
PostOfficeBox	String	0 to 1	The number of the address' Post Office Box.
AdditionalInformation	String	0 to 1	A place to specify information needed by more complex addresses.
City	String	1	The municipality in which the address exists.

Attribute/Role Name	Type	Multi- plicity	Description
State	String	1	The territorial division within the country in which the city exist.
Country	String	0 to 1	The nation in which the city exists.
PostalCode	String	0 to 1	The alphanumeric code assigned to the address to speed postal delivery.

6.1.2.1.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>CommunicationMethod</i>	Description.

6.1.2.1.3 Constraints and Notes

Constraint: In valid Address class instances, one and only one of the attributes Street and PostOfficeBox shall be present.

6.1.2.2 AreaType Class

The AreaType class provides a means to specify a quantity that represents the extent of a two-dimensional area and the unit associated with that area.

6.1.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Unit	AreaUnit	0 to 1	The area unit name.
Value	Decimal	1	The measured extent of an area.

6.1.2.2.2 Superclasses & Inherited Attributes

None.

6.1.2.2.3 Constraints and Notes

Note: If the Unit attribute is not present, the unit is specified by the value of the AreaUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the AreaUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the AreaUnit attribute in the UnitDefaults class definition.

6.1.2.3 BoundaryDefinition Class

The BoundaryDefinition class defines boundary, coordinate system, and unit information for a LayoutElement.

6.1.2.3.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi-plicity	Description
Unit	LayoutLengthUnit	0 to 1	The unit of measure for distances in this layout element.
CoordinateSystem	CoordinateSystem	0 to 1	The coordinate system for the LayoutElement.

6.1.2.3.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
SpatialDimension	Width, Depth, Height.

6.1.2.3.3 Constraints and Notes

Note: If the Unit attribute is not present, the default value for this attribute is “pixel”. This relationship is different than for other classes that have a Unit attribute.

Note: If the CoordinateSystem attribute is not present, the default value of this attribute is “upperLeftBased”.

6.1.2.4 ColorHighlight Class

The ColorHighlight class provides a means to specify how a shape’s color should be represented in a layout.

6.1.2.4.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi-plicity	Description
LineColor	ColorDefinition	0 to 1	The color to be used for the lines or edges that are a part of a shape.
FillColor	ColorDefinition	0 to 1	The color to be used for the inside area of a shape.
AlphaValue	NonNegativeInteger	0 to 1	A value between 0 and 100 defining the opaqueness of a shape.
Property	Property	0 to *	Name and value information for a noteworthy characteristic of a shape.

6.1.2.4.2 Superclasses & Inherited Attributes

None.

6.1.2.4.3 Constraints and Notes

Constraint: In a valid ColorHighlight instance, at least one of the attributes LineColor, FillColor, and AlphaValue shall appear.

6.1.2.5 CommunicationMethod Class

The *CommunicationMethod* class is abstract and provides a means to specify how a person or organization may be contacted.

6.1.2.5.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
Description	String	0 to 1	Additional information about a <i>CommunicationMethod</i> instance.

6.1.2.5.2 Superclasses & Inherited Attributes

None.

6.1.2.5.3 Constraints and Notes

None.

6.1.2.6 ContactInformation Class

The *ContactInformation* class provides a means to specify contact information about a person or organization.

6.1.2.6.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
ContactParty	ContactParty	0 to *	The person or organization to be contacted.
(subclass of) CommunicationMethod	(subclass of) CommunicationMethod	0 to *	The means by which the contact party can be communicated with.

6.1.2.6.2 Superclasses & Inherited Attributes

None.

6.1.2.6.3 Constraints and Notes

Constraint: In a valid *ContactInformation* instance, at least one attribute shall appear.

Note: The *CommunicationMethod* class is abstract, and as such, attributes with this name and type cannot be directly defined for a CMSD class. Any concrete subclass of the *CommunicationMethod* class can be used to define the name and type of an attribute for the *ContactInformation* class.

6.1.2.7 ContactParty Class

The *ContactParty* class provides a means to define the name of a person or organization that may be contacted in reference to some related production activity.

6.1.2.7.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PersonName	String	0 to 1	The name of a person that is associated with a production activity.
OrganizationName	String	0 to 1	The name of an organization that is associated with a production activity.

6.1.2.7.2 Superclasses & Inherited Attributes

None.

6.1.2.7.3 Constraints and Notes

Constraint: In a valid ContactParty instance, at least one of the attributes PersonName and OrganizationName shall appear.

6.1.2.8 Coordinate2D Class

The Coordinate2D class defines a point in a two-dimensional space.

6.1.2.8.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
X	Decimal	1	The distance along the x-axis for the point.
Y	Decimal	1	The distance along the y-axis for the point.

6.1.2.8.2 Superclasses & Inherited Attributes

None.

6.1.2.8.3 Constraints and Notes

None.

6.1.2.9 Coordinate3D Class

The Coordinate3D class defines a point in a three-dimensional space.

6.1.2.9.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Z	Decimal	0 to 1	The distance along the z-axis for the point.

6.1.2.9.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
Coordinate2D	X, Y.

6.1.2.9.3 Constraints and Notes

None.

6.1.2.10 CostAllocationData Class

The CostAllocationData class provides a means to specify information about the cost of performing a production activity or the cost for using a resource.

6.1.2.10.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
CostCategory	CostCategory	0 to 1	A general category in which costs can be grouped.
CostType	CostType	1	An indication of whether the cost is fixed or variable.
CostName	String	0 to 1	The name assigned to the cost item.
CostDescription	String	0 to 1	A description of the cost item.
VariableCostData	VariableCostData	0 to 1	Additional information about costs that are variable.
TotalCost	CurrencyType	1	The total amount of money associated with this cost item.
Property	Property	0 to *	Name and value information for a noteworthy characteristic associated with this cost item.

6.1.2.10.2 Superclasses & Inherited Attributes

None.

6.1.2.10.3 Constraints and Notes

Constraint: The VariableCostData attribute shall not appear in a CostAllocationData instance if the value of the CostType attribute is not "variable".

6.1.2.11 CurrencyType Class

The CurrencyType class provides a means to specify an amount of currency and the name of the currency unit associated with that amount.

6.1.2.11.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Unit	CurrencyUnit	0 to 1	The name of the currency unit.
Value	Decimal	1	An amount of currency.

6.1.2.11.2 Superclasses & Inherited Attributes

None.

6.1.2.11.3 Constraints and Notes

Note: If the Unit attribute is not present, the unit is specified by the value of the CurrencyUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the CurrencyUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the CurrencyUnit attribute in the UnitDefaults class definition.

6.1.2.12 Distribution Class

The Distribution class provides a means to define a statistical distribution.

6.1.2.12.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Name	String	1	The name of the statistical distribution.
Description	String	0 to 1	A description of the statistical distribution.
DistributionParameter	DistributionParameter	1 to *	The name, description, and value for information that describes a characteristic of the distribution.

6.1.2.12.2 Superclasses & Inherited Attributes

None.

6.1.2.12.3 Constraints and Notes

None.

6.1.2.13 DistributionDefinition Class

The DistributionDefinition class defines a statistical distribution that can be referenced.

6.1.2.13.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Distribution	Distribution	1	A definition of a statistical distribution.

6.1.2.13.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.1.2.13.3 Constraints and Notes

None.

6.1.2.14 DistributionParameter Class

The DistributionParameter class provides a means to specify information that defines the characteristics of a distribution.

6.1.2.14.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Name	String	1	The name of the statistical distribution parameter.
Description	String	0 to 1	A description of the statistical distribution parameter.
Value	Decimal	1	Data specifying the value of the distribution parameter.

6.1.2.14.2 Superclasses & Inherited Attributes

None.

6.1.2.14.3 Constraints and Notes

None.

6.1.2.15 Duration Class

The Duration class provides a means to define a period of time using either a fixed value or a statistical distribution.

6.1.2.15.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
---------------------	------	-------------------	-------------

Attribute/Role Name	Type	Multi- plicity	Description
Unit	TimeUnit	0 to 1	The unit of time measurement used for this duration instance.
Value	Decimal	0 to 1	The amount of time for a duration specified as a constant.
Distribution	Distribution	0 to 1	A statistical distribution defining the amount of time for the duration.
DistributionReference	DistributionDefinitionReference	0 to 1	A reference to a statistical distribution that defines the amount of time for the duration.

6.1.2.15.2 Superclasses & Inherited Attributes

None.

6.1.2.15.3 Constraints and Notes

Constraint: One and only one of the attributes Value, Distribution, and DistributionReference shall appear in a valid Duration class instance.

Note: If the Unit attribute is not present, the unit is specified by the value of the TimeUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the TimeUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the TimeUnit attribute in the UnitDefaults class definition

6.1.2.16 ElapsedTimeType Class

The ElapsedTimeType class provides a means to specify an amount of time and the name of the time unit associated with that amount.

6.1.2.16.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Unit	TimeUnit	0 to 1	The name of the time unit.
Value	Decimal	1	An amount of elapsed time.

6.1.2.16.2 Superclasses & Inherited Attributes

None.

6.1.2.16.3 Constraints and Notes

Note: If the Unit attribute is not present, the unit is specified by the value of the TimeUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the TimeUnit attribute does not appear in the

UnitDefaults instance, the unit is the default value specified for the TimeUnit attribute in the UnitDefaults class definition.

6.1.2.17 Email Class

The Email class provides a means to specify an email address that can be used to communicate with a person or organization.

6.1.2.17.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Address	URI	1	The email address needed to contact the person or organization.

6.1.2.17.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>CommunicationMethod</i>	Description.

6.1.2.17.3 Constraints and Notes

None.

6.1.2.18 Event Class

The Event class provides a means to plan for or record the occurrence of some phenomenon, condition, or state that is relevant to production activities.

6.1.2.18.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
SequenceNumber	String	0 to 1	A number that can be used to distinguish one event from another event.
Name	String	0 to 1	The name of the event.
Description	String	0 to 1	A description of the event.
Timestamp	Timestamp	0 to 1	The time that the event occurred or will occur.
Property	Property	0 to *	Name and value information for a noteworthy characteristic of the event.

6.1.2.18.2 Superclasses & Inherited Attributes

None.

6.1.2.18.3 Constraints and Notes

Constraint: A valid Event class instance shall have at least one attribute present.

6.1.2.19 GrossDimensions Class

The GrossDimensions class provides a means to specify the general dimension of an object and the length unit associated with the object's dimensions.

6.1.2.19.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Unit	LengthUnit	0 to 1	The name of the distance unit associated with the dimensions of the object.
Width	Decimal	0 to 1	A linear distance along an object's x-axis.
Depth	Decimal	0 to 1	A linear distance along an object's y-axis.
Height	Decimal	0 to 1	A linear distance along an object's z-axis.

6.1.2.19.2 Superclasses & Inherited Attributes

None.

6.1.2.19.3 Constraints and Notes

Constraint: Valid instances of the GrossDimensions class shall contain at least one of the attributes Width, Depth, and Height.

Note: CMSD does not specify a method for determining how an object should be oriented. The GrossDimensions class only provides a means to record the dimensions of an object after such a determination has been made.

Note: If the Unit attribute is not present, the unit is specified by the value of the LengthUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the LengthUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the LengthUnit attribute in the UnitDefaults class definition.

6.1.2.20 ImageResolution Class

The ImageResolution class provides a means to specify information about the resolution of an image graphic.

6.1.2.20.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PixelsPerUnit	NonNegativeInteger	1	The number of pixels in an image that represents a screen unit.

Attribute/Role Name	Type	Multi- plicity	Description
ScreenUnit	LayoutLengthUnit	1	A mapping of the pixels in an image to either real world or screen units.

6.1.2.20.2 Superclasses & Inherited Attributes

None.

6.1.2.20.3 Constraints and Notes

None.

6.1.2.21 ItemPackaging Class

The ItemPackaging class provides a means to specify information about how a part is packaged for sale or manufacture.

6.1.2.21.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PackageName	String	1	The name of this kind of packaging.
PackageDescription	String	0 to 1	A description of the packaging.
PerPiecePackaging	PerPiecePackaging	0 to 1	Information about packaging that is based on the number of pieces in the package.
PerMeasuredAmount Packaging	PerMeasuredAmountPackaging	0 to 1	Information about packaging that is based on the amount of a substance that is contained in the package.

6.1.2.21.2 Superclasses & Inherited Attributes

None.

6.1.2.21.3 Constraints and Notes

Constraint: In a valid ItemPackaging instance, at most one of the attributes PerPiecePackaging and PerMeasuredAmountPackaging shall appear.

6.1.2.22 JobConstraint Class

The JobConstraint class defines precedence constraint information for two jobs.

6.1.2.22.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PredecessorJob	JobReference	1	The job that is constraining the successor job.
SuccessorJob	JobReference	1	The job that is being constrained by the predecessor job.

6.1.2.22.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>PrecedenceConstraint</i>	Description, PrecedenceRelationship, LagDuration, LagPercentage, LagPartsComplete.

6.1.2.22.3 Constraints and Notes

None.

6.1.2.23 LengthType Class

The LengthType class provides a means to specify a linear distance quantity and the name of the distance unit associated with that quantity.

6.1.2.23.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Unit	LengthUnit	0 to 1	The name of the distance unit.
Value	Decimal	1	A quantity of linear distance.

6.1.2.23.2 Superclasses & Inherited Attributes

None.

6.1.2.23.3 Constraints and Notes

Note: If the Unit attribute is not present, the unit is specified by the value of the LengthUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the LengthUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the LengthUnit attribute in the UnitDefaults class definition.

6.1.2.24 LocationDefinition Class

The LocationDefinition class provides a means to specify general information about where a part or resource exists within a manufacturing enterprise.

6.1.2.24.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
FacilityLocation	String	0 to 1	Information about the facility where the part or resource resides.
WithinFacilityLocation	String	0 to 1	Information describing a place within a facility where a part or resource resides.
ResourceLocation	ResourceReference	0 to 1	A reference to a resource where the part or another resource resides.

6.1.2.24.2 Superclasses & Inherited Attributes

None.

6.1.2.24.3 Constraints and Notes

Constraint: In a valid LocationDefinition instance, at least one of its attributes shall appear.

6.1.2.25 LotInformation Class

The LotInformation class provides a means to specify information about a group of parts that were manufactured together and that have identifying characteristics that allow them to be distinguished from other groups of parts.

6.1.2.25.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
LotNumber	String	1	Identifying information for the lot.
ParentLotNumber	String	0 to 1	Information that indicates that this lot is a sub portion of a larger lot.
Description	String	0 to 1	Information about the distinguishing characteristics of this lot.

6.1.2.25.2 Superclasses & Inherited Attributes

None.

6.1.2.25.3 Constraints and Notes

None.

6.1.2.26 MachineProgramData Class

The MachineProgramData class provides a means to specify information about a machine program that will be used as a part of the process required to produce a part.

6.1.2.26.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
MachineProgramName	String	0 to 1	The name of the machine program.
TargetMachine	ResourceReference	0 to *	The machine or machines on which the machine program can run.
MachineProgramLocation	URI	0 to 1	The address from which the machine program can be located.

6.1.2.26.2 Superclasses & Inherited Attributes

None.

6.1.2.26.3 Constraints and Notes

Constraint: A valid MachineProgramData class instance shall have at least one attribute present.

6.1.2.27 MaintenanceProcessConstraint Class

The MaintenanceProcessConstraint class defines precedence constraint information for two maintenance processes.

6.1.2.27.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PredecessorProcess	MaintenanceProcessReference	1	The process that is constraining the successor process.
SuccessorProcess	MaintenanceProcessReference	1	The process that is being constrained by the predecessor process.

6.1.2.27.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>PrecedenceConstraint</i>	Description, PrecedenceRelationship, LagDuration, LagPercentage, LagPartsComplete.

6.1.2.27.3 Constraints and Notes

None.

6.1.2.28 PartGroup Class

The PartGroup class provides a means to specify information about a number of identical parts or a group of part instances that have been aggregated for some purpose.

6.1.2.28.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Description	String	0 to 1	Descriptive information about this group of parts.
PartType	PartTypeReference	0 to 1	The part type of the parts in this part group.
PartQuantity	NonNegativeInteger	0 to 1	The number of parts of the specified part type in the group.
PartInstance	PartReference	0 to *	A reference to a Part instance.

6.1.2.28.2 Superclasses & Inherited Attributes

None.

6.1.2.28.3 Constraints and Notes

Constraint: In a valid PartGroup instance, either the PartQuantity attribute shall appear or one or more PartInstance attributes shall appear.

Constraint: In a valid PartGroup instance, if the PartQuantity attribute appears then the PartType attribute shall also appear.

Note: In a valid PartGroup instance, it is valid for both the PartQuantity attribute and one or more PartInstance attributes to appear.

6.1.2.29 PerMeasuredAmountPackaging Class

The PerMeasuredAmountPackaging class provides a means to specify information about packaging for items that contain a specified amount of a substance.

6.1.2.29.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
MeasuredAmount	Decimal	1	The amount of the substance in the package.
PackageUnitOfMeasure	String	0 to 1	The unit of measure associated with the package.
PricePerUnit	CurrencyType	0 to 1	The price for a unit of the substance.

6.1.2.29.2 Superclasses & Inherited Attributes

None.

6.1.2.29.3 Constraints and Notes

None.

6.1.2.30 PerPiecePackaging Class

The PerPiecePackaging class provides a means to specify information about packaging for items that contain a group of individual pieces.

6.1.2.30.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PiecesPerPackage	NonNegativeInteger	1	The number of pieces in the package.
PricePerPiece	CurrencyType	0 to 1	The price for each piece in the package.

6.1.2.30.2 Superclasses & Inherited Attributes

None.

6.1.2.30.3 Constraints and Notes

None.

6.1.2.31 Phone Class

The Phone class provides a means to specify a phone number to be used to communicate with a person or organization.

6.1.2.31.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Number	String	1	The number needed to contact the person or organization by phone.

6.1.2.31.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>CommunicationMethod</i>	Description.

6.1.2.31.3 Constraints and Notes

None.

6.1.2.32 PowerType Class

The PowerType class provides a means to specify an amount of power and the name of the power unit associated with that amount.

6.1.2.32.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
---------------------	------	-------------------	-------------

Attribute/Role Name	Type	Multi- plicity	Description
Unit	PowerUnit	0 to 1	The name of the power unit.
Value	Decimal	1	An amount of power.

6.1.2.32.2 Superclasses & Inherited Attributes

None.

6.1.2.32.3 Constraints and Notes

Note: If the Unit attribute is not present, the unit is specified by the value of the PowerUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the PowerUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the PowerUnit attribute in the UnitDefaults class definition.

6.1.2.33 PrecedenceConstraint Class

The *PrecedenceConstraint* class provides a means to define information about the relationship between the initiation and completion of two production activities (jobs, processes, or maintenance processes).

6.1.2.33.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Description	String	0 to 1	A description of the constraint relationship between the two activities.
PrecedenceRelationship	PrecedenceRelationship	1	Whether the relationship between the two activities is SS, SF, FS, or FF.
LagDuration	ElapsedTimeType	0 to 1	A modification to the initiation or completion time of the predecessor activity.
LagPercentage	Decimal	0 to 1	A modification to the amount of work necessary to indicate the completion of the predecessor activity.
LagPartsComplete	NonNegativeInteger	0 to 1	A modification to the number of parts that will have to be completed to indicate the completion of the predecessor activity.

6.1.2.33.2 Superclasses & Inherited Attributes

None.

6.1.2.33.3 Constraints and Notes

None.

6.1.2.34 ProcessConstraint Class

The ProcessConstraint class defines precedence constraint information for two processes.

6.1.2.34.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
PredecessorProcess	ProcessReference	1	The process that is constraining the successor process.
SuccessorProcess	ProcessReference	1	The process that is being constrained by the predecessor process.

6.1.2.34.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>PrecedenceConstraint</i>	Description, PrecedenceRelationship, LagDuration, LagPercentage, LagPartsComplete.

6.1.2.34.3 Constraints and Notes

None.

6.1.2.35 Property Class

The Property class provides a means to specify a name, description, and additional data for a “property”. A property is a noteworthy characteristic of the CMSD entity defined by a CMSD class. An attribute of a CMSD class whose type is defined by this class is referred to as a “property attribute”.

6.1.2.35.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
PropertyDescription	PropertyDescriptionReference	0 to 1	A PropertyDescription class instance that describes the format and acceptable value space for a Property attribute.
Name	String	1	The name of a property.
Description	String	0 to 1	A description of the characteristic, trait, feature, or fact that this property denotes.
Unit	String	0 to 1	The unit associated with this property.

6.1.2.35.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
-----------------	----------------------

<i>ReferenceProperty</i>	An <i>AbstractEntityReference</i> subclass.
<i>SimpleProperty</i>	Value.
<i>StochasticProperty</i>	Distribution.

6.1.2.35.3 Constraints and Notes

- Constraint: If PropertyDescription information is provided, the other attributes that appear in a Property instance shall adhere to the unit, content, and cardinality constraints defined in the associated PropertyDescription instance.
- Constraint: The Value, Distribution, and “*AbstractEntityReference*” subclass attributes shall not appear together in a valid Property class instance.

6.1.2.36 QuantityType Class

The QuantityType class provides a means to specify the quantity of an object or of a measurable substance, where the unit associated with that object or substance is custom-defined.

6.1.2.36.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi-plicity	Description
UnitDescription	String	0 to 1	A description of the unit associated with the object or substance.
UnitName	String	1	A name for the unit associated with the object or substance.
Value	Decimal	1	An amount of an object or substance.

6.1.2.36.2 Superclasses & Inherited Attributes

None.

6.1.2.36.3 Constraints and Notes

None.

6.1.2.37 ReferenceMaterial Class

The ReferenceMaterial class provides a means to describe and provide a location for reference materials. Reference materials are existing organized aggregations of information (paper manuals, CAD files, applications on external media, web pages, etc.) that complement or supplement other information defined in CMSD.

6.1.2.37.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi-plicity	Description
---------------------	------	---------------	-------------

Attribute/Role Name	Type	Multi- plicity	Description
FileName	String	0 to 1	The name of a file containing reference material information.
OnlineLocation	URI	0 to 1	The online location or resource name of the reference material.
ISBN	String	0 to 1	The ISBN number associated with the reference material.
DigitalFormat	String	0 to 1	The format for a file that exists in electronic form.
PermanentStorageMedium	String	0 to 1	A specification of whether the reference material's information is stored as a hardcopy document, on a CD/DVD, in a computer file, etc.
PhysicalLocation	String	0 to 1	The location of a reference material that has a physical form.
RequiredApplication	RequiredApplication	0 to 1	Information about a computer application that is required to access the reference material's information.

6.1.2.37.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.1.2.37.3 Constraints and Notes

None.

6.1.2.38 *ReferenceProperty* Class

The *ReferenceProperty* class, a superclass of the *Property* class, provides a means for a *Property* class instance to define a reference to a CMSD entity instance.

6.1.2.38.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
(name of a subclass of) AbstractEntityReference	(type of a subclass of) AbstractEntityReference	0 to 1	A reference to an identifiable CMSD entity instance.

6.1.2.38.2 Superclasses & Inherited Attributes

None.

6.1.2.38.3 Constraints and Notes

Note: The *AbstractEntityReference* class is abstract, and as such, attributes with this name and type cannot be directly defined for a CMSD class. The name and type of the attribute for a concrete *ReferenceProperty* subclass instance is the name and type of the associated concrete *AbstractEntityReference* subclass instance.

6.1.2.39 RequiredApplication Class

The RequiredApplication class provides a means to specify information about a computer application that is required to access a reference material's information.

6.1.2.39.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
Name	String	1	The name of a computer application.
Description	String	0 to 1	The description of a computer application that is needed to access a reference material's information.
Version	String	0 to 1	The name/number of the specific version of a computer application.
OperatingSystem	String	0 to 1	The name of the computer operating system needed to run the application.

6.1.2.39.2 Superclasses & Inherited Attributes

None.

6.1.2.39.3 Constraints and Notes

None.

6.1.2.40 ResourcesRequired Class

The ResourcesRequired class provides a means to specify information about the characteristics of a group of resources that have been assembled for some purpose.

6.1.2.40.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Description	String	0 to 1	Information about the distinguishing characteristics of this group of resources.
ResourceClass	ResourceClassReference	0 to 1	The resource class for resources in the group.
MinimumNumber	NonNegativeInteger	0 to 1	The minimum number of resources that should be in this group.
MaximumNumber	NonNegativeInteger	0 to 1	The maximum number of resources that should be in this group.
Resource	ResourceReference	0 to *	A reference to a resource that is a part of this group.
AllowableSetup	SetupDefinitionReference	0 to *	A setup configuration supported by resources in this group.
RequiredEmployeeSkill	SkillReference	0 to *	A skill possessed by employee resources in this group.

6.1.2.40.2 Superclasses & Inherited Attributes

None.

6.1.2.40.3 Constraints and Notes

Constraint: In valid ResourcesRequired instances, at least one attribute appears.

Constraint: In valid ResourcesRequired instances, if the RequiredEmployeeSkill attribute is present and the ResourceClass attribute is present, the ResourceType attribute of the ResourceClass attribute shall be “employee”.

Constraint: In valid ResourcesRequired instances, if the AllowableSetup attribute is present and the ResourceClass attribute is present, the ResourceType attribute of the ResourceClass attribute shall not be “employee”.

Constraint: In valid ResourcesRequired instances, if the ResourceClass attribute is present, all resources referred to by a Resource attribute shall be of the the same ResourceType as the ResourceClass attribute.

6.1.2.41 ShapeLabelDefinition Class

The ShapeLabelDefinition class provides a means to define a text label to be associated with a shape.

6.1.2.41.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Text	String	1	The text content of the label.

Attribute/Role Name	Type	Multi- plicity	Description
Color	ColorDefinition	0 to 1	The color of the label.
Property	Property	0 to *	Name and value information for a noteworthy characteristic of the label.

6.1.2.41.2 Superclasses & Inherited Attributes

None.

6.1.2.41.3 Constraints and Notes

None.

6.1.2.42 SimpleProperty Class

The *SimpleProperty* class, a superclass of the Property class, provides a means for a Property class instance to define a simple value to be the content of a Property class instance.

6.1.2.42.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Value	String	0 to 1	Scalar data that defines the value of a property.

6.1.2.42.2 Superclasses & Inherited Attributes

None.

6.1.2.42.3 Constraints and Notes

None.

6.1.2.43 SpatialDimension Class

The SpatialDimension class defines the extent of the boundary of a space.

6.1.2.43.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Width	Decimal	1	The dimension of a space associated with the x-axis.
Depth	Decimal	1	The dimension of a space associated with the y-axis.
Height	Decimal	0 to 1	The dimension of a space associated with the z-axis.

6.1.2.43.2 Superclasses & Inherited Attributes

None.

6.1.2.43.3 Constraints and Notes

None.

6.1.2.44 SpeedType Class

The *SpeedType* class provides a means to specify the magnitude of a velocity and the name of the speed unit associated with that velocity.

6.1.2.44.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Unit	SpeedUnit	0 to 1	The name of the speed unit.
Value	Decimal	1	An amount of speed.

6.1.2.44.2 Superclasses & Inherited Attributes

None.

6.1.2.44.3 Constraints and Notes

Note: If the Unit attribute is not present, the unit is specified by the value of the SpeedUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the SpeedUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the SpeedUnit attribute in the UnitDefaults class definition.

6.1.2.45 StochasticProperty Class

The *StochasticProperty* class, a superclass of the Property class, provides a means for a Property class instance to indicate that a statistical distribution defines the value of a Property class instance.

6.1.2.45.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Distribution	Distribution	0 to 1	The statistical distribution that defines the value of a property.

6.1.2.45.2 Superclasses & Inherited Attributes

None.

6.1.2.45.3 Constraints and Notes

None.

6.1.2.46 TemperatureType Class

The TemperatureType class provides a means to specify a temperature magnitude and the name of the unit associated with that temperature.

6.1.2.46.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Unit	TemperatureUnit	0 to 1	The name of the temperature unit.
Value	Decimal	1	The temperature magnitude.

6.1.2.46.2 Superclasses & Inherited Attributes

None.

6.1.2.46.3 Constraints and Notes

Note: If the Unit attribute is not present, the unit is specified by the value of the TemperatureUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the TemperatureUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the TemperatureUnit attribute in the UnitDefaults class definition.

6.1.2.47 VariableCostData Class

The VariableCostData class provides a means to specify information about cost items whose total cost is proportional to the amount of production activity taking place.

6.1.2.47.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
CostUnit	String	1	The name of the metric by which costs can be measured.
CostRatePerUnit	Decimal	1	The amount of cost incurred for each cost unit.
CostUnitQuantity	Decimal	1	The number of cost units assigned to the associated cost item.

6.1.2.47.2 Superclasses & Inherited Attributes

None.

6.1.2.47.3 Constraints and Notes

None.

6.1.2.48 VolumeType Class

The VolumeType class provides a means to specify a quantity that represents the magnitude of a volume and the unit associated with that volume.

6.1.2.48.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Unit	VolumeUnit	0 to 1	The volume unit name.
Value	Decimal	1	The measured extent of the volume.

6.1.2.48.2 Superclasses & Inherited Attributes

None.

6.1.2.48.3 Constraints and Notes

Note: If the Unit attribute is not present, the unit is specified by the value of the VolumeUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the VolumeUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the VolumeUnit attribute in the UnitDefaults class definition.

6.1.2.49 WeightType Class

The WeightType class provides a means to specify an amount of weight and the name of the weight unit associated with that amount.

6.1.2.49.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Unit	WeightUnit	0 to 1	The name of the weight unit.
Value	Decimal	1	An amount of weight.

6.1.2.49.2 Superclasses & Inherited Attributes

None.

6.1.2.49.3 Constraints and Notes

Note: If the Unit attribute is not present, the unit is specified by the value of the WeightUnit attribute in the UnitDefaults instance defined in the associated CMSDDocument instance. If the UnitDefaults instance is not present or the WeightUnit attribute does not appear in the UnitDefaults instance, the unit is the default value specified for the WeightUnit attribute in the UnitDefaults class definition.

6.2 Basic Types Package

The Basic Types package contains definitions for basic data types that are used to define the format and value space for CMSD class attributes. Diagrams and descriptions of each class defining a basic type are provided in the sections below.

6.2.1 Diagrams

6.2.1.1 Extended Primitive Data Types

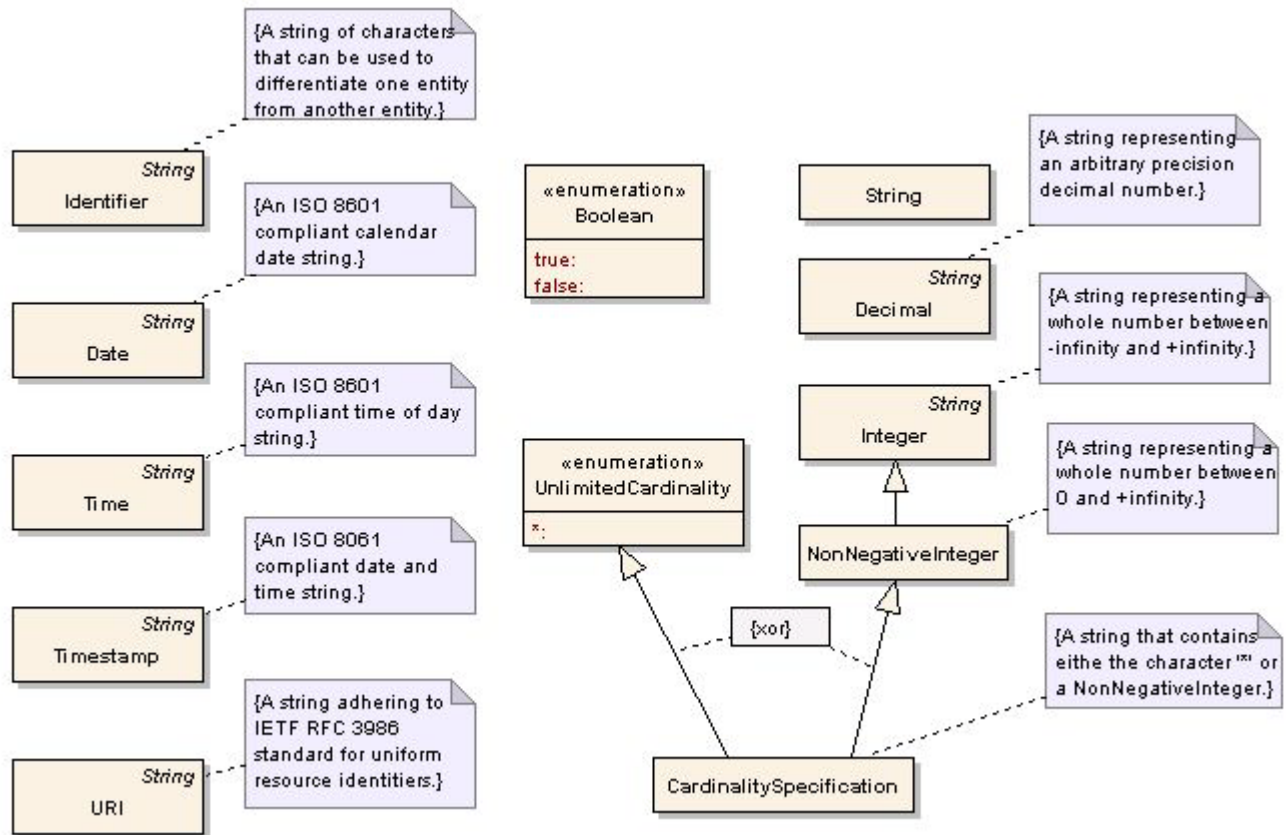


Figure 12: Extended Primitive Data Types

6.2.1.2 Types Related to Values for Units of Measure

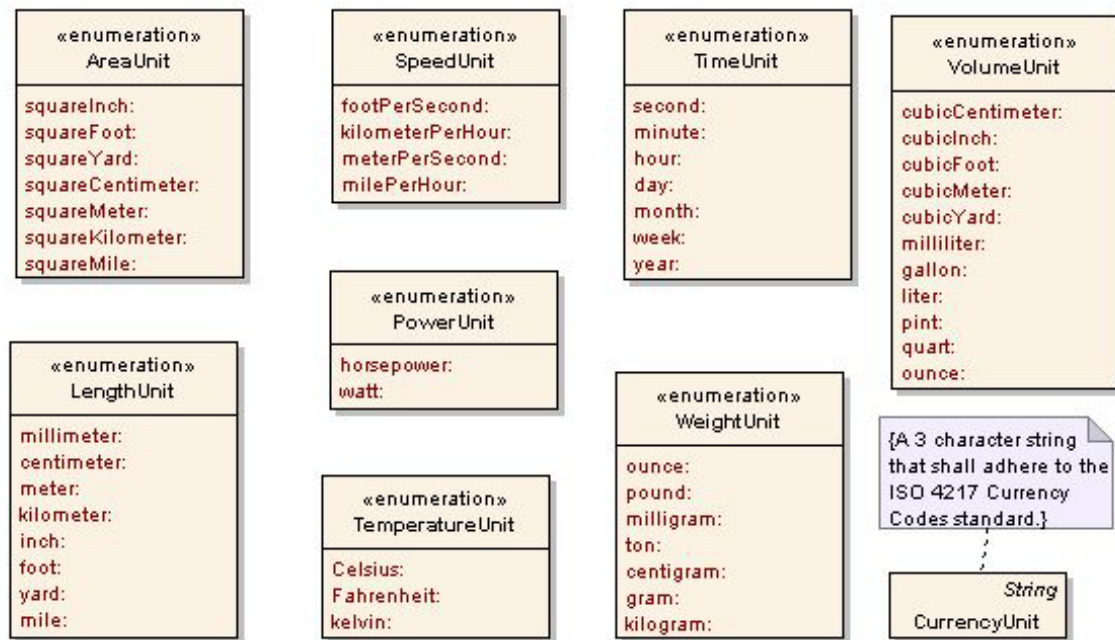


Figure 13: Types Related to Values for Units of Measure

6.2.1.3 Types Supporting the Definition of Metadata Information



Figure 14: Types Supporting the Definition of Metadata Information

6.2.1.4 Types Supporting Layout Information Definition

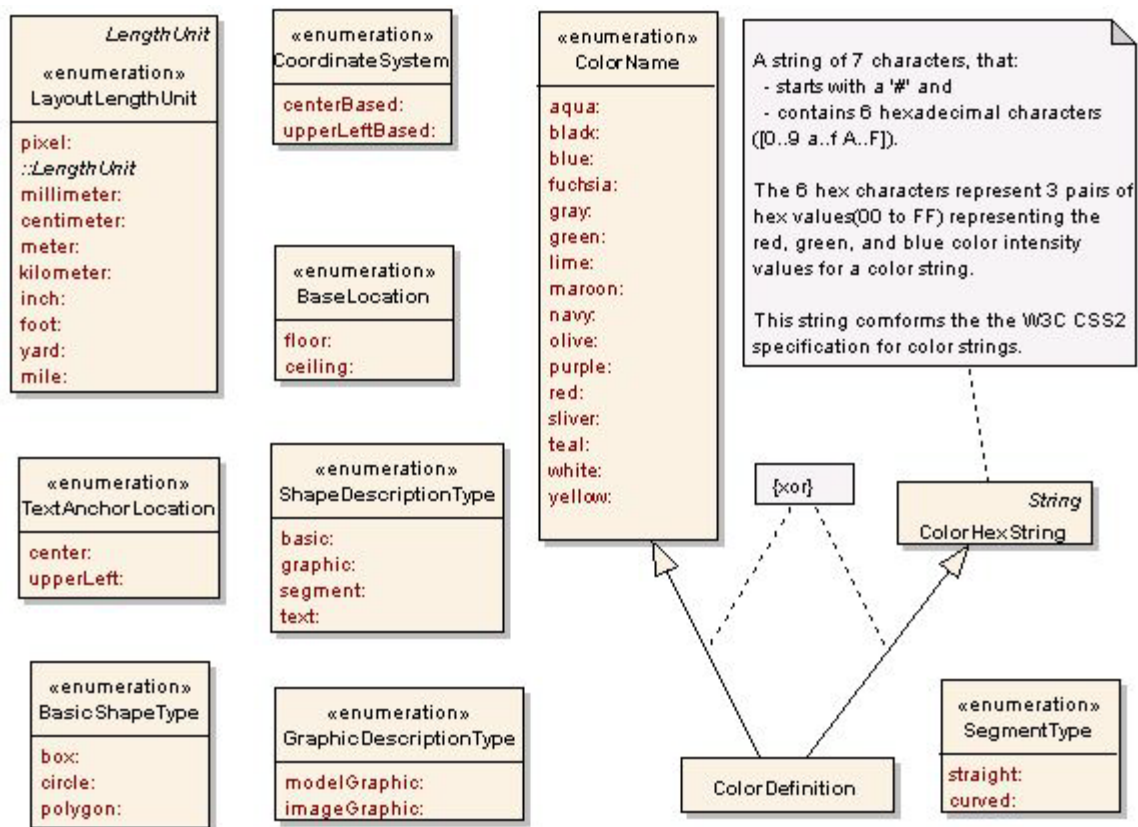


Figure 15: Types Supporting Layout Information Definition

6.2.1.5 Other Basic Data Types

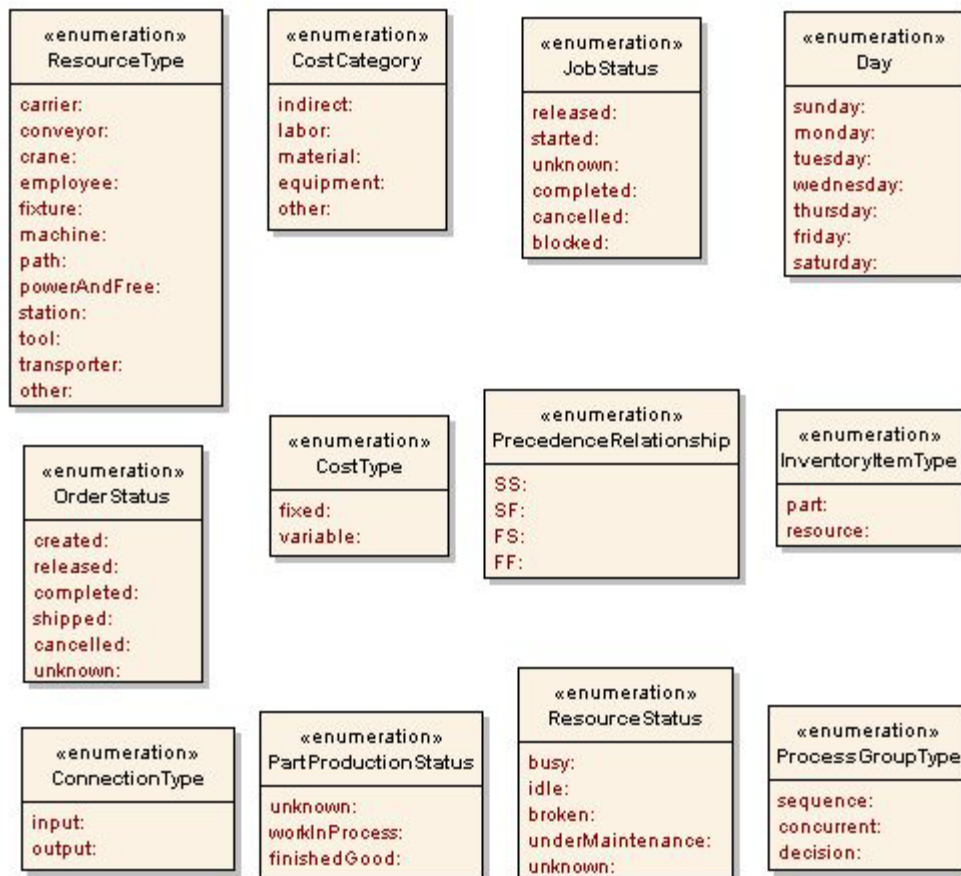


Figure 16: Classes Defining Other Basic Data Types

6.2.2 Classes

6.2.2.1 AreaUnit Class

The AreaUnit class is a data type that defines enumeration values representing the names for units of measure related to area measurements. The relationship between the AreaUnit literals and standardized measures of area are specified in NIST Special Publication 330, which is based on ISO 31. The set of valid enumeration literals for the AreaUnit class is a subset of the allowed names for area units defined in those standards.

The AreaUnit class and the enumeration literals it defines are presented in Figure 13.

6.2.2.2 BaseLocation Class

The BaseLocation class is a data type that defines enumeration values that specify the plane within the space defined by a LayoutObject instance on which the bottom of a shape associated with that LayoutObject instance should be attached. “floor” indicates that shapes should be attached to the plane at the bottom of the space. “ceiling” indicates that shapes should be attached to the plane at the top of the space. In either

case, the body of the shape should be oriented so that it remains within the space defined by the `LayoutObject` instance.

The `BaseLocation` class and the enumeration literals it defines are presented in Figure 15.

6.2.2.3 BasicShapeType Class

The `BasicShapeType` class is a data type that defines enumeration values that specify whether the basic shape used to define the shape of a layout object will be a box, circle, or polygon shape.

The `BasicShapeType` class and the enumeration literals it defines are presented in Figure 15.

6.2.2.4 Boolean Class

The `Boolean` class is a data type that defines enumeration values that express the result of a logical operation. The enumeration value “true” denotes a true logical operation result and the value “false” denotes a false logical operation result.

The `Boolean` class and the values associated with it are presented in Figure 12.

6.2.2.5 CardinalitySpecification Class

The `CardinalitySpecification` class defines a data type that represents a string that may contain either a non negative integer or the enumeration value “*”. This class is a subclass of both the `UnlimitedCardinality` class and the `NonNegativeInteger` class.

The `CardinalitySpecification` class is presented in Figure 12.

6.2.2.6 ColorDefinition Class

The `ColorDefinition` class represents a string that is either a color name as defined by the `ColorName` class, or a hexadecimal representation of the RGB light intensity values that define a color as defined by the `ColorHexString` class.

The `ColorDefinition` class is presented in Figure 15.

6.2.2.7 ColorHexString Class

The `ColorHexString` class, a subclass of the `String` class, represents a string that holds a hex representation for the RGB light intensity values associated with a color. The format of the string is the character “#” followed by six characters chosen from the characters “0” through “9”, “a” through “f”, and “A” through “F”. The format of this string conforms to the W3C CSS2 specification.

The `ColorHexString` class is presented in Figure 15.

6.2.2.8 ColorName Class

The `ColorName` class is a data type that defines enumeration values for the names for colors as defined in the W3C “HTML 4.01 Specification”. The RGB light intensity values associated with each color name are also defined in that standard.

The `ColorName` class and the enumeration literals it defines are presented in Figure 15.

6.2.2.9 ConnectionType Class

The `ConnectionType` class is a data type that defines enumeration values representing the two types of connections (input connections and output connections) that may apply to a resource group member.

The ConnectionType class and the enumeration literals it defines are presented in Figure 16.

6.2.2.10 CoordinateSystem Class

The CoordinateSystem class is a data type that defines enumeration values that indicate the location of the origin and the orientation of the axes of a layout's coordinate system. The origin is the point in the space defined for a layout where x, y, and z are all 0. Below is a description of each coordinate system from the perspective of an external observer positioned above the space.

For a "centerBased" coordinate system, the origin is at the center of the space. The x-axis should be thought of as horizontal with positive values increasing towards the right and negative values increasing towards the left. The y-axis should be thought of as vertical with positive values increasing in the "upwards" direction and negative values increasing in the "downwards" direction. The z-axis should be thought of as perpendicular to both the x-axis and y-axis with positive values increasing in direction towards the observer and negative values increasing in the direction away from the observer. .

For an "upperLeftBased" coordinate system the origin is at the upper left bottom of the space. Only non-negative values are defined for the x, y, and z axes. The x-axis should be thought of as horizontal with positive values increasing towards the right. The y-axis should be thought of as vertical with positive values increasing in the "downwards" direction. The z-axis should be thought of as perpendicular to both the x-axis and y-axis with positive values increasing towards the observer.

The CoordinateSystem class and the enumeration literals it defines are presented in Figure 15.

6.2.2.11 CostCategory Class

The CostCategory class is a data type that defines enumeration values representing different kinds of costs that may be incurred due to the execution of production activities.

The CostCategory class and the enumeration literals it defines are presented in Figure 16.

6.2.2.12 CostType Class

The CostType class is a data type that defines enumeration values indicating whether an expense incurred because of a production activity is fixed or variable.

The CostType class and the enumeration literals it defines are presented in Figure 16.

6.2.2.13 CurrencyUnit Class

The CurrencyUnit class, a subclass of the String class, defines a data type that represents the code associated with an international currency. Valid values for class attributes of this type shall adhere to the 3 character code representation for currency defined in ISO 4217. The list of valid values is defined in that standard.

The CurrencyUnit class is presented in Figure 13.

6.2.2.14 Date Class

The Date class, a subclass of the String class, defines a data type that represents a specific day of a year. Valid values for class attributes of this type shall adhere to the representation for calendar date defined in ISO 8061. The format for valid values is **YYYY-MM-DD**, where YYYY is the year in the Gregorian calendar, MM is the month, and DD is the day in the month.

The Date class is presented in Figure 12.

6.2.2.15 Day Class

The Day class is a data type that defines enumeration values representing the names of the seven days of the week.

The Day class and the enumeration literals it defines are presented in Figure 16.

6.2.2.16 Decimal Class

The Decimal class, a subclass of the String class, defines a data type that represents a decimal number of arbitrary precision. Data specified as being of this type shall adhere to the definition of the decimal primitive datatype defined in section 3.2.1 of the W3C recommendation, "XML Schema Part 2: Datatypes".

The Decimal class is presented in Figure 12.

6.2.2.17 GraphicDescriptionType Class

The GraphicDescriptionType class is a data type that defines enumeration values that indicate that a GraphicDescription subclass instance refers to data for a 2D or 3D model of an object (modelGraphic) or a 2D image (imageGraphic).

The GraphicDescriptionType class and the enumeration literals it defines are presented in Figure 15.

6.2.2.18 Identifier Class

The Identifier class, a subclass of the String class, defines a data type representing the value of a CMSD "identifier". Class attributes of this type provide a means for the instances of its parent class to be differentiated from each other, or a means to refer to specific instances of other classes.

The Identifier class is presented in Figure 12.

6.2.2.19 Integer Class

The Integer class, a subclass of the String class, defines a data type that represents a whole number whose value is between -infinity and +infinity. Data specified as being of this type shall adhere to the definition of the integer primitive datatype defined in section 3.2.1 of the W3C recommendation, "XML Schema Part 2: Datatypes".

The Integer class is presented in Figure 12.

6.2.2.20 InventoryItemType Class

The InventoryItemType class is a data type that defines enumeration values representing the two broad categories of things that may be tracked in inventory, "part" and "resource".

The InventoryItemType class and the enumeration literals it defines are presented in Figure 16.

6.2.2.21 JobStatus Class

The JobStatus class is a data type that defines enumeration values representing the state of a job.

The JobStatus class and the enumeration literals it defines are presented in Figure 16.

6.2.2.22 LengthUnit Class

The LengthUnit class is a data type that defines enumeration values representing the names for units of measure related to distance measurements. The relationship between the LengthUnit literals and

standardized measures of distance are specified in NIST Special Publication 330, which is based on ISO 31. The set of valid enumeration literals for the LengthUnit class is a subset of the allowed names for distance units defined in those standards.

The LengthUnit class and the enumeration literals it defines are presented in Figure 13.

6.2.2.23 LayoutLengthUnit Class

The LayoutLengthUnit class is a subclass of the LengthUnit class. It extends the definition of that class with the additional literal value “pixel”. This provides a means to support the definition of layout information that is based on screen units and not real-world distance units.

The LayoutLengthUnit class and the enumeration literals it defines are presented in Figure 15.

6.2.2.24 NonNegativeInteger Class

The NonNegativeInteger class, a subclass of the Integer class, defines a data type that represents a whole number whose value is between 0 and +infinity.

The NonNegativeInteger class is presented in Figure 12.

6.2.2.25 OrderStatus Class

The OrderStatus class is a data type that defines enumeration values representing the state of an order.

The OrderStatus class and the enumeration literals it defines are presented in Figure 16.

6.2.2.26 PartProductionStatus Class

The PartProductionStatus class is a data type that defines enumeration values representing the current state of the part.

The PartProductionStatus class and the enumeration literals it defines are presented in Figure 16.

6.2.2.27 PowerUnit Class

The PowerUnit class is a data type that defines enumeration values representing the names for units of measurement relating to the specification of a rate of mechanical output. The relationship between the PowerUnit literals and standardized measures of power are specified in NIST Special Publication 330, which is based on ISO 31. The set of valid values for the PowerUnit class is a subset of the allowed names for power units defined in those standards.

The PowerUnit class and the enumeration literals it defines are presented in Figure 13.

6.2.2.28 PrecedenceRelationship Class

The PrecedenceRelationship class is a data type that defines enumeration values that specify initiation and completion constraints between production activities that are either planned or currently under way. A relationship between two production activities involving a PrecedenceRelationship always takes the following form:

“predecessor activity” PrecedenceRelationship “successor activity”

The predecessor activity and successor activity shall either be a job, a process, or a maintenance process. The table below specifies how a precedence relationship between two activities should be interpreted.

Precedence Relationship	Meaning
SS	The predecessor activity cannot start until the successor activity starts.
SF	The predecessor activity cannot start until the successor activity finishes.
FS	The predecessor activity cannot finish until the successor activity starts.
FF	The predecessor activity cannot finish until the successor activity finishes.

The PrecedenceRelationship class and the enumeration literals it defines are presented in Figure 16.

6.2.2.29 ProcessGroupType Class

The ProcessGroupType class is a data type that defines enumeration values representing the different kinds of process groups. In a “sequence” process group all of the processes in the group are to be executed sequentially. In a “concurrent” process group all of the processes may be executed concurrently. In a “decision” process group a single process in the group is selected for execution based on information defined with the group.

The ProcessGroupType class and the enumeration literals it defines are presented in Figure 16.

6.2.2.30 PropertyExtensibleEntity Class

The PropertyExtensibleEntity class is a data type that defines enumeration values for the names of each of the CMSD classes that have Property attributes. The enumeration values defined in this class support the definition of metadata to extend the kinds of information that can be represented in CMSD.

The PropertyExtensibleEntity class and the enumeration literals it defines are presented in Figure 14.

6.2.2.31 PropertyType Class

The PropertyType class is a data type that defines enumeration values representing the different kinds of Property attributes. A “simple” Property attribute contains information whose type is defined by the SimpleDataTypeName class. A “reference” Property attribute contains information that refers to a CMSD entity that is defined elsewhere in a CMSDDocument instance or in a separate CMSDDocument instance. A “stochastic” Property attribute contains information defining a statistical distribution.

The PropertyType class and the enumeration literals it defines are presented in Figure 14.

6.2.2.32 ReferenceTypeName Class

The ReferenceTypeName class is a data type that defines enumeration values for the names of each of the CMSD classes that defines information allowing different CMSD entities to refer to each other. The enumeration values defined in this class support the definition of metadata to extend the kinds of information that can be represented in CMSD.

The ReferenceTypeName class and the enumeration literals it defines are presented in Figure 14.

6.2.2.33 ResourceStatus Class

The ResourceStatus class is a data type that defines enumeration values representing the operational state of a resource.

The ResourceStatus class and the enumeration literals it defines are presented in Figure 16.

6.2.2.34 ResourceType Class

The ResourceType class is a data type that defines enumeration values representing the different kinds of resources that may be modeled with the Resource class.

The ResourceType class and the enumeration literals it defines are presented in Figure 16.

6.2.2.35 SegmentType Class

The SegmentType class is a data type that defines enumeration values that indicate whether a segmented shape is straight or curved.

The SegmentType class and the enumeration literals it defines are presented in Figure 15.

6.2.2.36 ShapeDescriptionType Class

The ShapeDescriptionType class is a data type that defines enumeration values that indicate that a ShapeDescription instance defines a basic shape, a graphic image or model, a segmented shape, or a text annotation.

The ShapeDescriptionType class and the enumeration literals it defines are presented in Figure 15.

6.2.2.37 SimpleDataTypeName Class

The SimpleDataTypeName class is a data type that defines enumeration values for the names of each of the CMSD classes that may be used as the type for a “simple” Property attribute. The enumeration values defined in this class support the definition of metadata to extend the kinds of information that can be represented in CMSD.

The SimpleDataTypeName class and the enumeration literals it defines are presented in Figure 14.

6.2.2.38 SpeedUnit Class

The SpeedUnit class is a data type that defines enumeration values representing the names for units of measure related to velocity measurements. The relationship between the SpeedUnit literals and standardized measures of velocity are specified in NIST Special Publication 330, which is based on ISO 31. The set of valid values for the SpeedUnit class is a subset of the allowed names for velocity units defined in those standards.

The SpeedUnit class and the enumeration literals it defines are presented in Figure 13.

6.2.2.39 String Class

The String class defines a data type that is a finite sequence of characters. Data specified as being of this type shall adhere to the definition of the string primitive datatype defined in section 3.2.1 of the W3C recommendation, “XML Schema Part 2: Datatypes”.

The String class is presented in Figure 12.

6.2.2.40 TemperatureUnit Class

The TemperatureUnit class is a data type that defines enumeration values representing the names for units of measure related to temperature measurements. The relationship between the TemperatureUnit literals and standardized measures of temperature are specified in NIST Special Publication 330, which is based on

ISO 31. The set of valid values for the TemperatureUnit class is a subset of the allowed names for temperature units defined in those standards.

The TemperatureUnit class and the enumeration literals it defines are presented in Figure 13.

6.2.2.41 TextAnchorLocation Class

The TextAnchorLocation class is a data type that defines enumeration values indicating how a text string should be attached to a layout coordinate. The enumeration value “center” specifies that the center of the text string should be attached at the specified layout coordinate. The enumeration value “upperLeft” specifies that the upper left corner of the text string should be attached at the specified layout coordinate.

The TextAnchorLocation class and the enumeration literals it defines are presented in Figure 15.

6.2.2.42 Time Class

The Time class, a subclass of the String class, defines a data type that represents the time of day. Valid values for class attributes of this type shall adhere to the representation for time of day defined in ISO 8061. The format for valid values is hh:mm:ss, where hh represents hours, mm represents minutes, and ss represents seconds.

The Time class is presented in Figure 12.

6.2.2.43 Timestamp Class

The Timestamp class, a subclass of the String class, defines a data type that represents a specific instant in time. Valid values for class attributes of this type shall adhere to the representation for date and time defined in ISO 8061. The format for valid values is YYYY-MM-DDThh:mm:ss (YYYY-MM-DD is a valid date, followed by the character “T”, followed by a valid time string of the format hh:mm:ss.)

The Timestamp class is presented in Figure 12.

6.2.2.44 TimeUnit Class

The TimeUnit class is a data type that defines enumeration values representing the names associated with discrete time-periods. The definition of the time-period associated with the “second”, “minute”, “hour”, and “day” TimeUnit enumeration literals are defined in ISO 1000. The time-periods associated with the week, month, and year enumeration literals in this class represent approximate time periods, and the actual time-period is dependent on the situation in which enumeration literal is used.

The TimeUnit class and the enumeration literals it defines are presented in Figure 13.

6.2.2.45 UnitTypeName Class

The UnitTypeName class is a data type that defines enumeration values for the names of each of the CMSD classes that defines information about units of measure. The enumeration values defined in this class support the definition of metadata to extend the kinds of information that can be represented in CMSD.

The UnitTypeName class and the enumeration literals it defines are presented in Figure 14.

6.2.2.46 UnlimitedCardinality Class

The UnlimitedCardinality class is a data type that defines the enumeration value “*”. This class is a superclass of the CardinalitySpecification class. The enumeration value “*” denotes a non-negative integer whose exact value is not specified.

The UnlimitedCardinality class is presented in Figure 12.

6.2.2.47 URI Class

The URI class, subclass of the String class, defines a data type that represents a Uniform Resource Identifier (URI), which is a sequence of characters that identifies or names a resource on the internet. The syntax for this type is defined by Internet Engineering Task Force (IETF) Network Working Group Request For Comments 3986 (RFC 3986) standard. URIs represent a collection of common syntax schemes used to identify information on the Internet, such as Uniform Resource Locators (URLs) (e.g., “http://www.ietf.org/rfc/rfc3986.txt”) and Uniform Resource Names (URNs) (e.g., “urn:example:test1”).

The URI class is presented in Figure 12.

6.2.2.48 VolumeUnit Class

The VolumeUnit class is a data type that defines enumeration values representing the names for units of measure related to volume measurements. The relationship between the VolumeUnit literals and standardized measures of volume are specified in NIST Special Publication 330, which is based on ISO 31. The set of valid values for the VolumeUnit class is a subset of the allowed names for volume units defined in those standards.

The VolumeUnit class and the enumeration literals it defines are presented in Figure 13.

6.2.2.49 WeightUnit Class

The WeightUnit class is a data type that defines enumeration values representing the names for units of measure related to mass measurements. The relationship between the WeightUnit literals and standardized measures of weight (mass) are specified in NIST Special Publication 330, which is based on ISO 31. The set of valid values for the WeightUnit class is a subset of the allowed names for mass units defined in those standards.

The WeightUnit class and the enumeration literals it defines are presented in Figure 13.

6.3 CMSD Package

The CMSD package defines packages that contain definitions for all of the classes and relationships that make up the CMSD information model. Although it does not directly define any classes, several nested sub-packages are defined within the scope of the CMSD package. It is within these nested packages that the classes and relationships that define CMSD information are directly defined. Each of the CMSD package's subpackages defines a focused, cohesive set of classes and relationships for a specific subset of CMSD information.

6.3.1 Diagrams

6.3.1.1 Top Level CMSD Packages

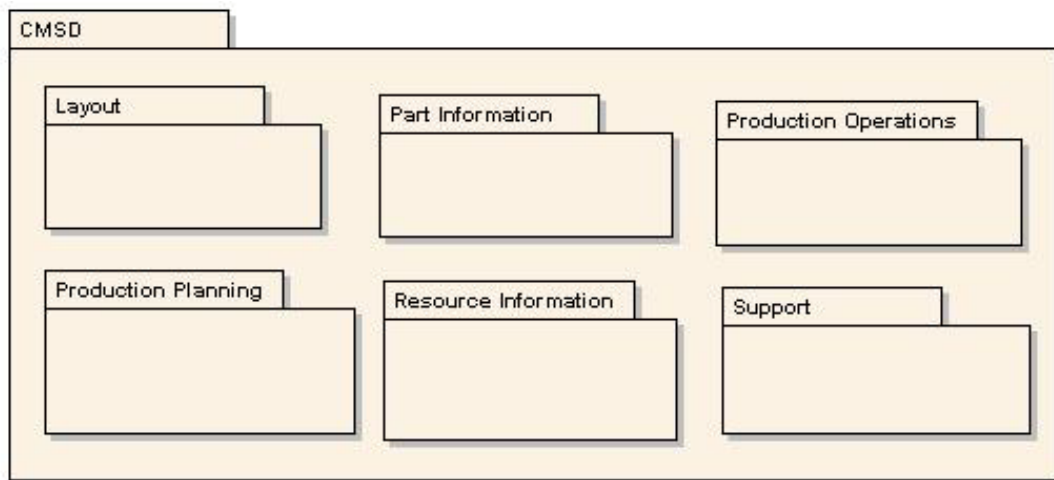


Figure 17: Top Level CMSD Packages

The Layout package defines classes and relationships that facilitate the creation of manufacturing layout information. A manufacturing layout is a specification of the spatially-oriented characteristics and interrelationships for the logical and physical entities that are used to carry out production activities.

The Part Information package defines classes and relationships for describing the raw materials, work-in-progress components, and finished products that are the inputs to and outputs of manufacturing processes. In addition, information about the component structure of parts and about the kinds and amounts of parts either completed or available to be used in production activities can also be defined.

The Production Operations package defines classes and relationships describing customer requests for products, production requests to produce those products, and the effort needed to produce those products.

The Production Planning package contains classes and relationships that can be used to create plans describing the dates and hours of operation for production resources and plans describing the sequence of processing steps necessary to manufacture products using the available production resources.

The Resource Information package contains classes for creating definitions of the characteristics and capabilities of the equipment and employees used in the manufacturing process, the skills associated with employees, and setup information required for the making efficient use of non-employee resources.

The Support package defines packages that contain definitions for simple types and other basic structures that are used in other CMSD packages to define more complex structures.

6.3.2 Classes

No classes are directly defined by the CMSD package. Each of the subpackages of the CMSD package will be described separately in other sections of this document.

6.4 Conceptual Framework Package

The Conceptual Framework package, a subpackage of the Support package, contains classes and relationships that define how important groupings of CMSD information, referred to as entities, relate to each other, and under what circumstances those entities may be considered unique.

6.4.1 Diagrams

6.4.1.1 Conceptual Framework Classes

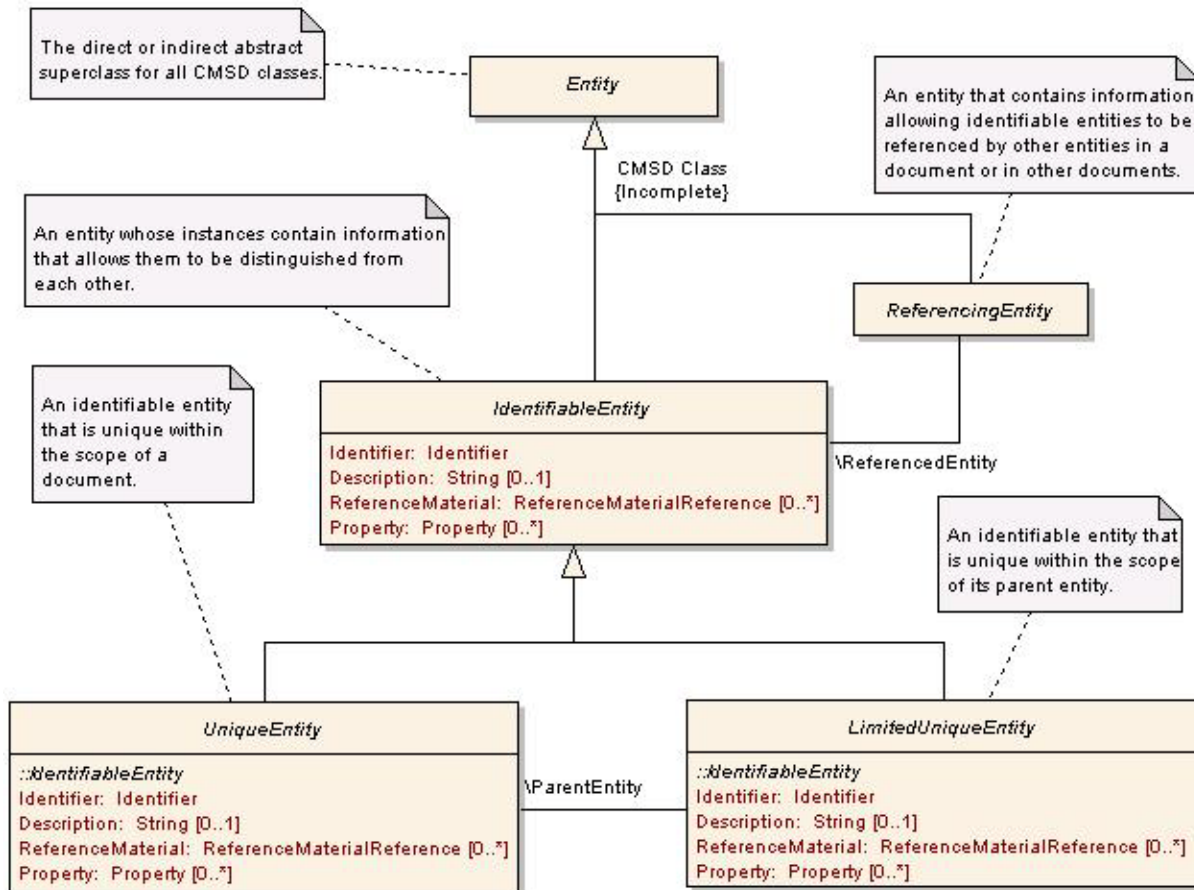


Figure 18: Classes of the Conceptual Framework Package

The *Entity* class is a generalization of all classes defined in CMSD. Conceptually, it is the ancestor of all CMSD classes. In the diagram above, this is indicated by the depiction of the *Entity* class as a generalization of classes in the set of all CMSD classes. Although only two direct subclasses are shown, the “incomplete” constraint indicates the existence of other subclasses of the *Entity* class that are not shown on this diagram.

The *IdentifiableEntity* class represents CMSD entities that are unique within a specified scope of a CMSD document. Instances of subclasses of *IdentifiableEntity* can be differentiated from each other by comparing the value of their Identifier attributes.

The *UniqueEntity* class, a subclass of the *IdentifiableEntity* class, represents a CMSD entity that is unique within the total scope of a CMSD document.

The *LimitedUniqueEntity* class, a subclass of the *IdentifiableEntity* class, represents a CMSD entity that is unique within the scope of an instance of a *UniqueEntity* subclass (referred to as the *ParentEntity*), but that might not be unique within the total scope of a CMSD document.

The *ReferencingEntity* class is a CMSD entity that defines information indicating that a given CMSD entity instance is in some way associated with or related to another CMSD entity instance that is defined in another part of a CMSD document.

6.4.2 Classes

6.4.2.1 Entity Class

The *Entity* class is a generalization of all classes defined in CMSD.

6.4.2.1.1 Defined Attributes and Association Roles

The *Entity* class does not define any attributes or have any explicitly defined association roles.

6.4.2.1.2 Superclasses & Inherited Attributes

None.

6.4.2.1.3 Constraints and Notes

Note: The *Entity* class is the ancestor of all other CMSD classes.

As depicted in Figure 18, the *Entity* class is a generalization for all classes in the generalization set “CMSD Class”. This indicates that the *Entity* class is implicitly the direct or indirect ancestor of all other classes in CMSD, whether depicted on other diagrams or not. The “incomplete” constraint on the explicit subclass relationship between the *Entity* class and *IdentifiableEntity* and *ReferencingEntity* classes further indicates that other direct subclasses of *Entity* exist.

To reduce model clutter, the generalization relationship between the *Entity* class and other CMSD classes that are direct subclasses of *Entity* will not be depicted on other diagrams. Direct generalization/specialization relationships between other CMSD classes will be depicted.

6.4.2.2 IdentifiableEntity Class

The *IdentifiableEntity* class defines the common properties of a CMSD entity that is unique within a specified scope of a CMSD document.

6.4.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Identifier	Identifier	1	A sequence of characters allowing instances of this class to be differentiated from each other.
Description	String	0 to 1	Information describing what this class represents.

Attribute/Role Name	Type	Multiplicity	Description
ReferenceMaterial	ReferenceMaterialReference	0 to *	An indication of the existence of information defined external to a CMSD document that may contain information relevant to the information defined by this class.
Property	Property	0 to *	Name and value information for a noteworthy characteristic of the entity defined by this class. The allowable format and value space for a Property can be defined by creating a PropertyDescription and associating the Property with it.

6.4.2.2.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>Entity</i>	

6.4.2.2.3 Constraints and Note

None.

6.4.2.3 LimitedUniqueEntity Class

The *LimitedUniqueEntity* represents a CMSD entity that is unique within the scope of the “ParentEntity” that contains it.

6.4.2.3.1 Defined Attributes and Association Roles

None.

6.4.2.3.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>IdentifiableEntity</i>	Identifier, Description, ReferenceMaterial, Property.

6.4.2.3.3 Constraints and Notes

Note: Derived association ParentEntity is conceptual only.

In Figure 18 the derived association ParentEntity is depicted. The ParentEntity association indicates a possibly indirect association with a *LimitedUniqueEntity* subclass instance and the *UniqueEntity* subclass instance that contains it.

This derived association is conceptual and is included to promote understanding of the model, but it does not indicate that an explicit attribute named “ParentEntity” exists in instances of the *LimitedUniqueEntity* class.

6.4.2.4 ReferencingEntity Class

The *ReferencingEntity* represents a CMSD entity that defines information allowing it to refer to an *IdentifiableEntity*.

6.4.2.4.1 Defined Attributes and Association Roles

None.

6.4.2.4.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>Entity</i>	

6.4.2.4.3 Constraints and Notes

Note: This class defines no attributes. Subclasses will define attributes to enable referencing of *IdentifiableEntity* instances.

Note: Derived association ReferencedEntity is conceptual only.

In Figure 18, the derived association ReferencedEntity is depicted. The ReferencedEntity association indicates an indirect association with an *IdentifiableEntity* subclass instance and the *ReferencingEntity* subclass instance that refers to it.

This derived association is conceptual and is included to promote understanding of the model, but it does not indicate that an explicit attribute named “ReferencedEntity” exists in instances of *ReferencingEntity* subclasses.

6.4.2.5 UniqueEntity Class

The *UniqueEntity* represents a CMSD entity that is unique within the scope of a CMSD document.

6.4.2.5.1 Defined Attributes and Association Roles

None.

6.4.2.5.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>IdentifiableEntity</i>	Identifier, Description, ReferenceMaterial, Property.

6.4.2.5.3 Constraints and Notes

None.

6.5 Document Definition Package

The Document Definition package defines classes and relationships that describe the structure and allowable content for a “CMSD document”. A CMSD document is an assemblage of data components, defined according to CMSD class and relationship definitions, that has been assembled with some specific purpose in mind, and that is suitable for archival or exchange.

6.5.1 Diagrams

6.5.1.1 Classes of the Document Definition Package

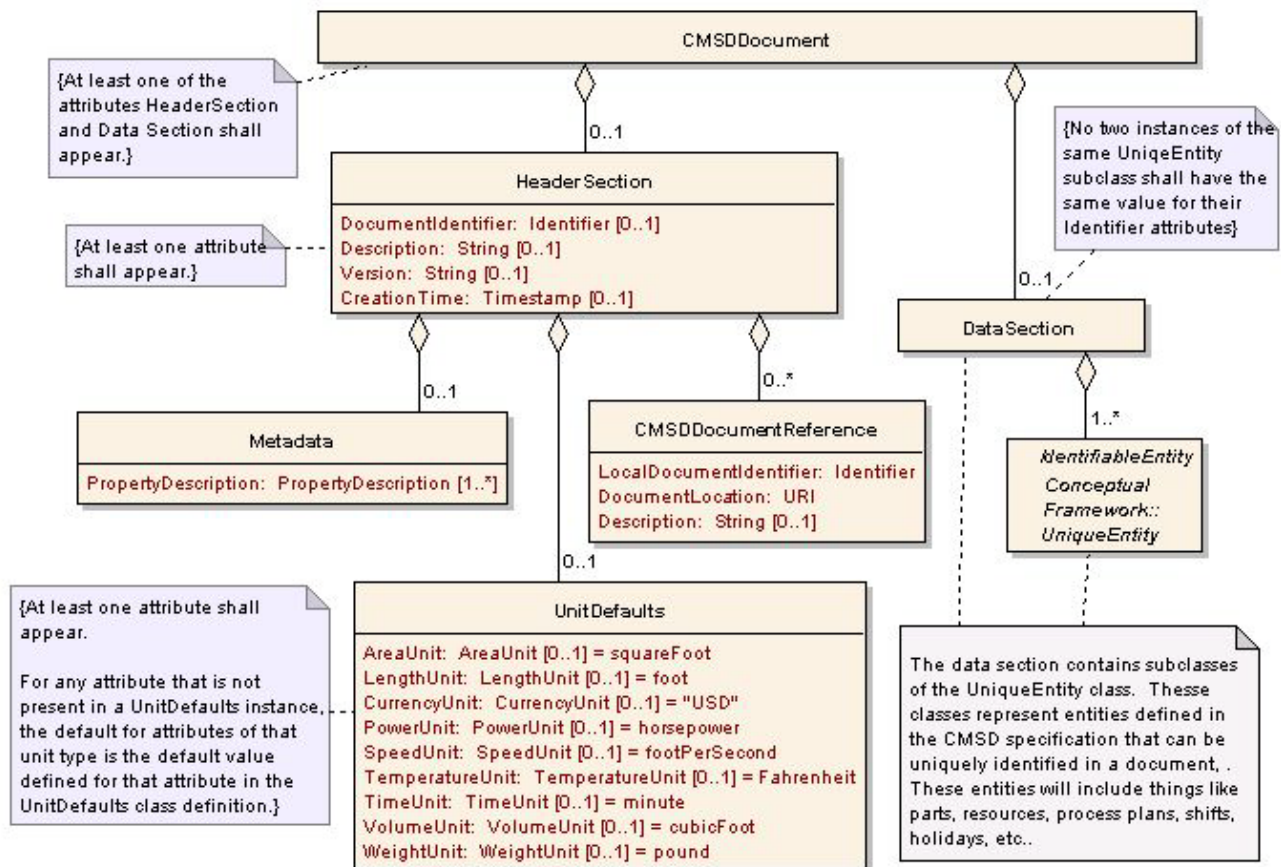


Figure 19: Classes of the Document Definition Package

The CMSDDocument class facilitates the creation of a collection of instances of CMSD classes. A CMSDDocument instance represents the scope for determining the uniqueness of the instances of the CMSD entities defined within it, and enables the ability to unambiguously reference CMSD information, whether defined within its scope or defined within the scope of another CMSDDocument instance.

The HeaderSection class defines information enabling a CMSDDocument instance to be differentiated from other CMSDDocument instances, metadata information describing properties that can be used to extend the definition of CMSD entities, and information about the default units of measure that are implicitly associated with CMSD information when unit information is not explicitly specified.

The DataSection class represents a collection of instances of uniquely identifiable CMSD entities. Generally, most of the information in a CMSDDocument instance will exist as CMSD Class instances that are nested within an instance of the DataSection class.

The Metadata class represents a collection of PropertyDescription class instances that define information about the allowable content and value space for Property attributes defined in CMSD classes.

The UnitDefaults class defines default values for the units of measure that are defined for some CMSD classes.

The CMSDDocumentReference class provides a means for information in another CMSDDocument instance to be referenced by entities in this CMSDDocument instance.

6.5.2 Classes

6.5.2.1 CMSDDocument Class

The CMSDDocument class defines the structure of a collection of CMSD entities that have been assembled with some specific purpose in mind, such that the assembled information is suitable for archiving or exchange between software applications.

6.5.2.1.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
HeaderSection	HeaderSection	0 to 1	Descriptive information about this document and its purpose, metadata information describing the meaning of Property attributes used to extend the definition of CMSD entities, information about the default values for units of measure, and information facilitating the referencing of other CMSD documents by this CMSD document.
DataSection	DataSection	0 to 1	A collection of uniquely identifiable CMSD class instances.

6.5.2.1.2 Superclasses & Inherited Attributes

None.

6.5.2.1.3 Constraints and Notes

Constraint: In valid instances of the CMSDDocument class, at least one of the attributes HeaderSection and DataSection shall appear.

6.5.2.2 CMSDDocumentReference Class

The CMSDDocumentReference class specifies the location of a separate CMSDDocument instance and the value of an identifier to be used to refer to information in that document from the current CMSDDocument instance.

6.5.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
LocalDocumentIdentifier	Identifier	1	The DocumentIdentifier that is used to indicate referenced information is defined in a separate CMSDDocument instance.
DocumentLocation	URI	1	The digital location of the separate CMSDDocument instance.
Description	String	0 to 1	Information about the separate CMSDDocument being referenced.

6.5.2.2.2 Superclasses & Inherited Attributes

None.

6.5.2.2.3 Constraints and Notes

Note: The document referenced by the value of the DocumentLocation attribute should be a valid CMSDDocument instance.

6.5.2.3 DataSection Class

The DataSection class defines information about a collection of uniquely identifiable CMSD Class instances.

6.5.2.3.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
(concrete subclass of) <i>UniqueEntity</i>	(concrete subclass of) <i>UniqueEntity</i>	1 to *	An instance of one of the concrete subclasses of the UniqueEntity class.

6.5.2.3.2 Superclasses & Inherited Attributes

None.

6.5.2.3.3 Constraints and Notes

Constraint: Within a DataSection instance, no two instances of the same *UniqueEntity* subclass shall have the same value for their Identifier attribute.

Note: *UniqueEntity* or any abstract subclass of *UniqueEntity* shall not appear as an attribute of a DataSection instance. Only Concrete subclasses of *UniqueEntity* shall appear as attributes of a DataSection instance. The name and type of the attribute will be the name and type of the concrete subclass.

6.5.2.4 HeaderSection Class

The HeaderSection class defines general information about the CMSDDocument instance, Property attribute metadata information, information about defaults for units, and information enabling the referencing of CMSD entity information defined in other CMSDDocument instances.

6.5.2.4.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
DocumentIdentifier	Identifier	0 to 1	The DocumentIdentifier that is used to indicate referenced information is in the current CMSDDocument instance.
Description	String	0 to 1	Information about this CMSDDocument instance.
Version	String	0 to 1	Information indicating this CMSDDocument is a variant of a related CMSDDocument.
CreationTime	Timestamp	0 to 1	The time and date that this CMSDDocument instance was created.
Metadata	Metadata	0 to 1	Information describing the allowable content and format for Property attributes.
UnitDefaults	UnitDefaults	0 to 1	Information specifying the implicit value for a unit of measure that is associated with an attribute when that attribute's unit value information is not explicitly specified.
CMSDDocumentReference	CMSDDocumentReference	0 to *	Information specifying the location of a separate CMSDDocument instance and an identifier that can be used to refer to that document in the current CMSDDocument instance.

6.5.2.4.2 Superclasses & Inherited Attributes

None.

6.5.2.4.3 Constraints and Notes

Constraint: In a valid instance of the HeaderSection class, at least one of its attributes shall appear.

Note: If the UnitDefaults attribute is not present in a HeaderSection instance, the default for any attribute (in the current CMSDDocument instance) defined with the AreaUnit, CurrencyUnit, TimeUnit, LengthUnit, PowerUnit, SpeedUnit, TemperatureUnit, VolumeUnit, or WeightUnit types shall be the default defined in the UnitDefaults class definition for that type.

6.5.2.5 Metadata Class

The Metadata class defines a collection of PropertyDescription class instances that specify the content and format of CMSD entity Property attributes.

6.5.2.5.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PropertyDescription	PropertyDescription	1 to *	Information about allowable content and format for a CMSD entity Property attribute.

6.5.2.5.2 Superclasses & Inherited Attributes

None.

6.5.2.5.3 Constraints and Notes

None.

6.5.2.6 UnitDefaults Class

The UnitDefaults class defines the default unit value for an instance of any class that has an attribute whose type is defined by the AreaUnit, CurrencyUnit, LengthUnit, PowerUnit, SpeedUnit, TemperatureUnit, TimeUnit, VolumeUnit, or WeightUnit classes.

6.5.2.6.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
AreaUnit	AreaUnit	0 to 1	The default value for an area unit of measure.
CurrencyUnit	CurrencyUnit	0 to 1	The default value for a currency unit of measure.
LengthUnit	LengthUnit	0 to 1	The default value for a length unit of measure.
PowerUnit	PowerUnit	0 to 1	The default value for a power unit of measure.
SpeedUnit	SpeedUnit	0 to 1	The default value for a speed unit of measure.
TemperatureUnit	TemperatureUnit	0 to 1	The default value for a temperature unit of measure.
TimeUnit	TimeUnit	0 to 1	The default value for a time unit of measure.

Attribute/Role Name	Type	Multi- plicity	Description
VolumeUnit	VolumeUnit	0 to 1	The default value for a volume unit of measure.
WeightUnit	WeightUnit	0 to 1	The default value for a weight unit of measure.

6.5.2.6.2 Superclasses & Inherited Attributes

None.

6.5.2.6.3 Constraints and Notes

Constraint: At least one its attributes shall appear in a valid instance of the UnitDefaults class.

Note: If the attribute specifying the default value for a unit type is not present in a UnitDefaults instance, the default value for that unit type shall be specified by the default for the associated attribute in the UML UnitDefaults class definition. The UnitDefaults class definition is in Figure 19.

6.6 Entity Reference Definition Package

The Entity Reference Definition package defines classes and relationships for defining information that allows a CMSD entity to refer to another CMSD entity. There are many entity reference classes defined in this package, and they may be grouped into two categories, “basic” entity references and “complex” entity references.

Some CMSD entities may be referenced by providing the name of the entity to be referenced, identifier information about the CMSD document that contains the instance of the entity, and identifier information for the instance of the entity being referenced. These entities are referenced through the use of what are called basic entity references.

Other CMSD entities allow not only the main CMSD entity instance to be referenced but also locally unique identifiable information within the main CMSD entity instance to be referenced. Referencing information in an entity such as this requires the value for the identifier of the main CMSD entity and the values for the identifiers of the limited unique entity instances that are nested within the main CMSD entity.

All of the entity reference classes inherit from the AbstractEntityReference class. This class provides a means to specify information about the CMSD document in which an entity instance is defined.

6.6.1 Diagrams

6.6.1.1 The AbstractEntityReference Class and the Basic Entity Reference Classes

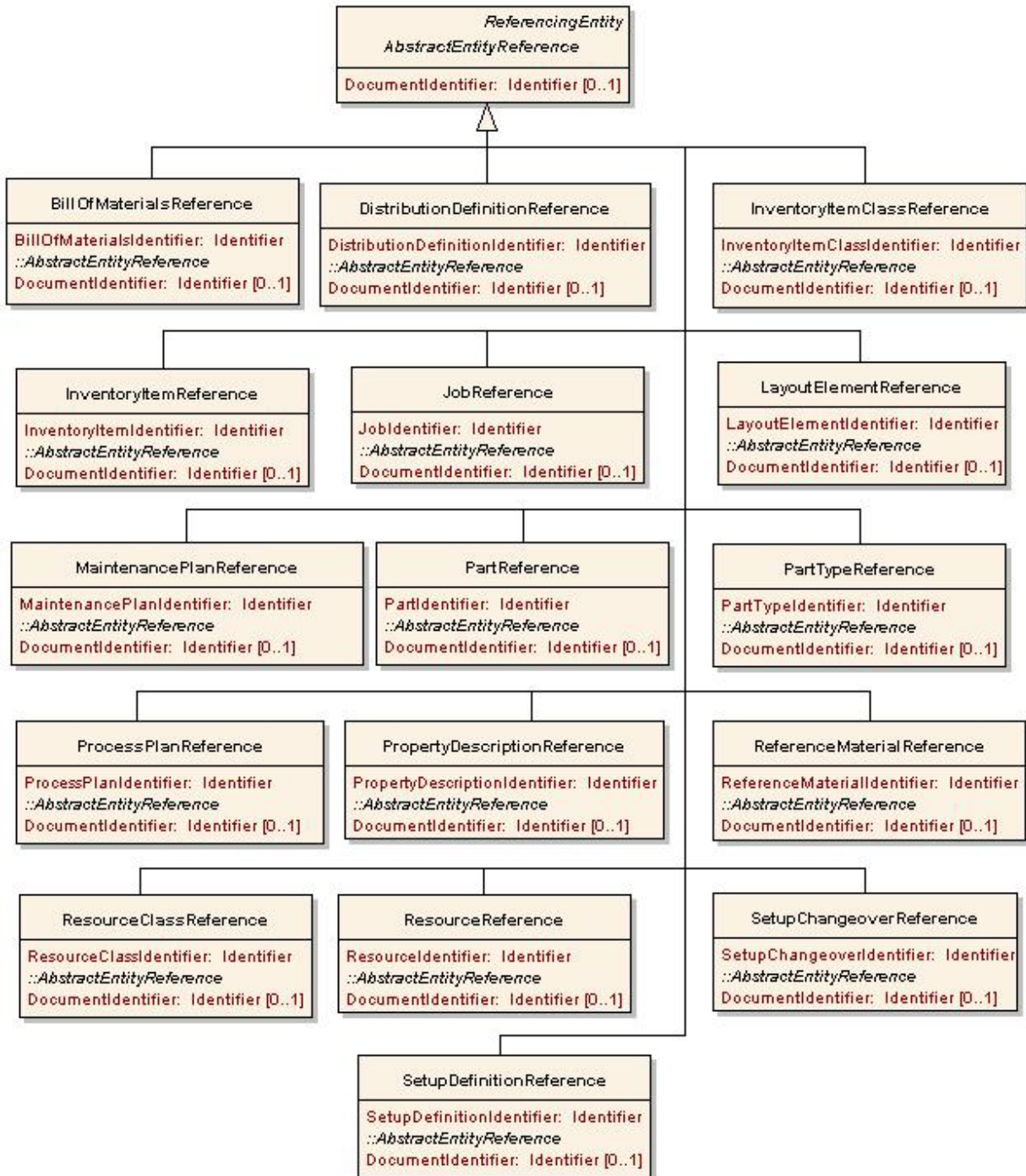


Figure 20: The AbstractEntityReference Class and the Basic Entity Reference Classes

The **AbstractEntityReference** class represents an abstract generalization of all of the entity reference classes. It defines information about the CMSD document in which a referenced entity instance is defined.

Basic entity reference is the name for a group of subclasses of the `AbstractEntityReference` class that provide a means to reference instances of subclasses of the `UniqueEntity` class. The following classes are basic entity reference classes:

<code>BillOfMaterialsReference</code> class	<code>DistributionDefinitionReference</code> class
<code>InventoryItemClassReference</code> class	<code>InventoryItemReference</code> class
<code>JobReference</code> class	<code>LayoutElementReference</code> class
<code>MaintenancePlanReference</code> class	<code>PartReference</code> class
<code>PartTypeReference</code> class	<code>ProcessPlanReference</code> class
<code>PropertyDescriptionReference</code> class	<code>ReferenceMaterialReference</code> class
<code>ResourceClassReference</code> class	<code>ResourceReference</code> class
<code>SetupChangeoverReference</code> class	<code>SetupDefinitionReference</code> class

For each of these classes, the name of the CMSD entity being referenced is the first part of the name of the class. Also each of these classes has an attribute to hold the value of the identifier of the CMSD entity instance being referenced, where the name of CMSD entity being referenced is also the first part of the name of the attribute.

6.6.1.2 Complex Entity Reference Classes

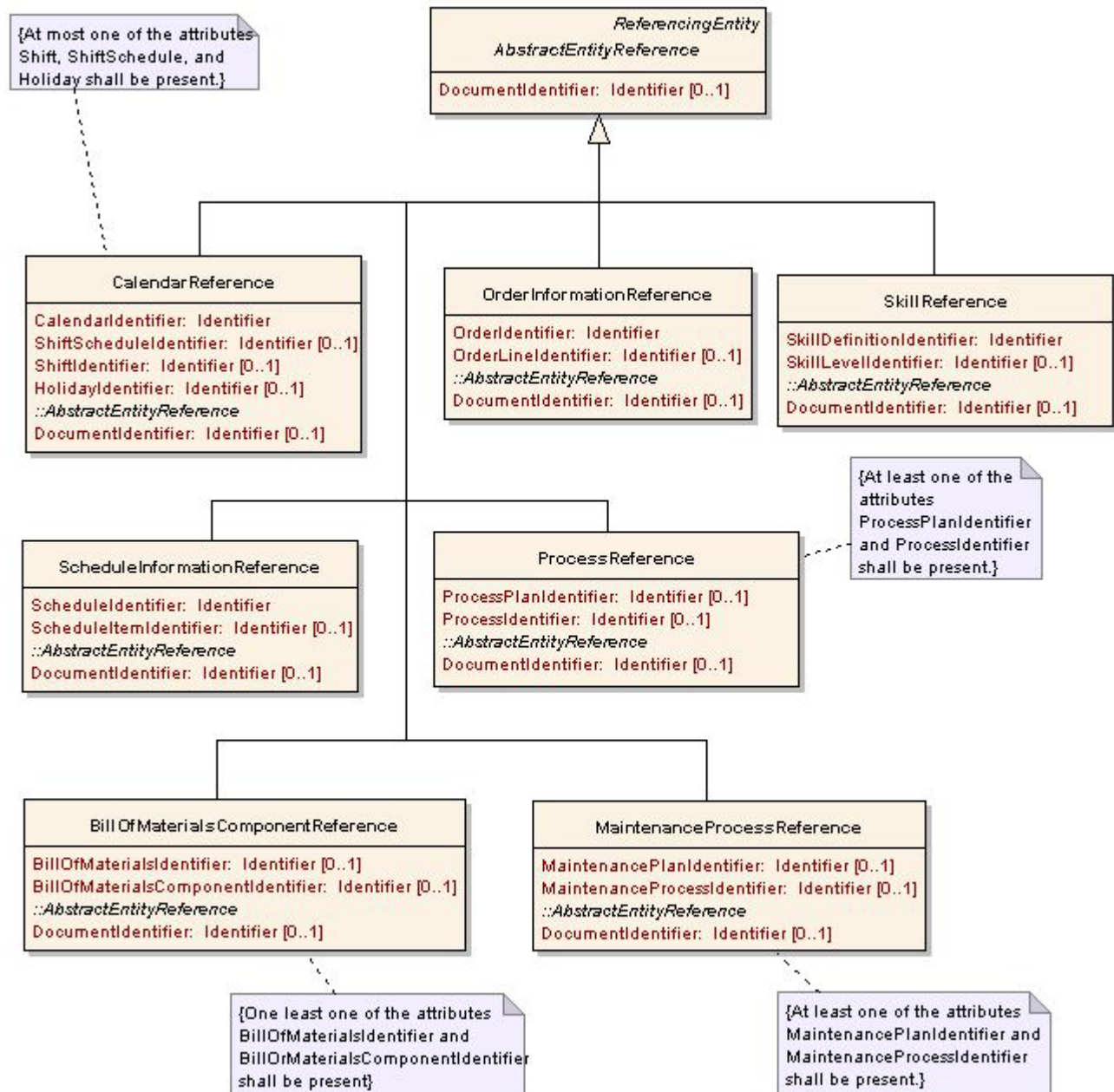


Figure 21: Complex Entity Reference Classes

The CalendarReference class provides a means to create a reference to a Calendar instance, and optionally to a ShiftSchedule, Shift, or Holiday instance within the Calendar instance.

The OrderInformationReference class provides a means to create a reference to an Order instance, and optionally to an OrderLine instance within that Order instance.

The SkillReference class provides a means to create a reference to a Skill instance, and optionally to a SkillLevel instance.

The *ScheduleInformationReference* class provides a means to create a reference to a *Schedule* instance or to a *ScheduleItem* instance within a specified *Schedule* instance.

The *ProcessReference* class provides a means to create a reference to a *Process* instance within a *ProcessPlan* instance.

The *BillOfMaterialsComponentReference* class provides a means to create a reference to a *BillOfMaterialsComponent* instance within a *BillOfMaterials* instance.

The *MaintenanceProcessReference* class provides a means to create a reference to a *MaintenanceProcess* instance within a *MaintenancePlan* instance.

6.6.2 Classes

6.6.2.1 *AbstractEntityReference* Class

The *AbstractEntityReference* class is an abstract generalization of all of the other entity reference classes, and provides a means to specify information about the CMSD document that contains the entity information being referenced.

6.6.2.1.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
DocumentIdentifier	Identifier	0 to 1	The identifier of the CMSDDocument instance that contains information that is being referenced.

6.6.2.1.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>ReferencingEntity</i>	None.

6.6.2.1.3 Constraints and Notes

None.

6.6.2.2 Basic Entity Reference Classes -

BillOfMaterialsReference, DistributionDefinitionReference, InventoryItemClassReference, InventoryItemReference, JobReference, LayoutElementReference, MaintenancePlanReference, PartReference, PartTypeReference, ProcessPlanReference, PropertyDescriptionReference, ReferenceMaterialReference, ResourceClassReference, ResourceReference, SetupChangeoverReference, and SetupDefinitionReference

The classes listed above all have the same general structure and each provides the ability to define reference information for a specific subclass of the *UniqueEntity* class. For each of these classes, the name of the class specifies the CMSD entity that it specifies reference information for. The format for the name of one of these classes is "EntityName" (the CMSD entity being referenced) with the word "Reference" appended as a suffix.

6.6.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
"EntityName" Reference	Identifier	1	The identifier value associated with a UniqueEntity subclass instance. In each basic entity class the actual name of the attribute with the identifier value will have the characters " <u>EntityName</u> " replaced by the name of the entity being referenced.

6.6.2.2.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>AbstractEntityReference</i> (subclass of <i>ReferencingEntity</i>)	DocumentIdentifier.

6.6.2.2.3 Constraints and Notes

Note: The CMSD class instance being referenced should be consistent with the name of the basic entity class that is referencing it.

6.6.2.3 BillOfMaterialsComponentReference Class

The BillOfMaterialsComponentReference class provides a means to create a reference to a BillOfMaterialsComponent class instance.

6.6.2.3.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
BillOfMaterialsIdentifier	Identifier	0 to 1	The identifier of the BillOfMaterials instance containing the BillOfMaterialsComponent instance being referenced.
BillOfMaterialsComponentIdentifier	Identifier	0 to 1	The identifier of a BillOfMaterialsComponent instance within the BillOfMaterials instance.

6.6.2.3.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>AbstractEntityReference</i> (subclass of <i>ReferencingEntity</i>)	DocumentIdentifier.

6.6.2.3.3 Constraints and Notes

Constraint: In valid instances of the *BillOfMaterialsComponentReference* class, at least one of the attributes *BillOfMaterialsIdentifier* and *BillOfMaterialsComponentIdentifier* shall be present.

Note: When the *BillOfMaterialsIdentifier* attribute appears without the *BillOfMaterialsComponentIdentifier* attribute, the interpretation is the reference is to all of the *BillOfMaterialsComponent* instances in the referenced *BillOfMaterials* instance. The interpretation is the same as if a *BillOfMaterialsReference* instance had been used.

6.6.2.4 CalendarReference Class

The *CalendarReference* class provides a means to create a reference to a *Calendar* instance, and optionally to a *ShiftSchedule*, *Shift*, or *Holiday* instance within the *Calendar* instance.

6.6.2.4.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
<i>CalendarIdentifier</i>	Identifier	1	The identifier of the <i>Calendar</i> class instance being referenced.
<i>ShiftScheduleIdentifier</i>	Identifier	0 to 1	The identifier of a <i>ShiftSchedule</i> class instance within the <i>Calendar</i> instance.
<i>ShiftIdentifier</i>	Identifier	0 to 1	The identifier of a <i>Shift</i> class instance within the <i>Calendar</i> instance.
<i>HolidayIdentifier</i>	Identifier	0 to 1	The identifier of a <i>Holiday</i> class instance within the <i>Calendar</i> instance.

6.6.2.4.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>AbstractEntityReference</i> (subclass of <i>ReferencingEntity</i>)	<i>DocumentIdentifier</i> .

6.6.2.4.3 Constraints and Notes

Constraint: Valid instances of the *CalendarReference* class shall not contain more than one of the attributes *Shift*, *ShiftSchedule*, and *Holiday*.

6.6.2.5 MaintenanceProcessReference Class

The *MaintenanceProcessReference* class provides a means to create a reference to a *MaintenanceProcess* instance.

6.6.2.5.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi-plicity	Description
MaintenancePlanIdentifier	Identifier	0 to 1	The identifier of the MaintenancePlan class instance containing the MaintenanceProcess instance being referenced.
MaintenanceProcessIdentifier	Identifier	0 to 1	The identifier of a MaintenanceProcess class instance within the MaintenancePlan instance.

6.6.2.5.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>AbstractEntityReference</i> (subclass of <i>ReferencingEntity</i>)	DocumentIdentifier.

6.6.2.5.3 Constraints and Notes

Constraint: In valid instances of the MaintenanceProcessReference class, at least one of the attributes MaintenancePlanIdentifier and MaintenanceProcessIdentifier shall be present.

Note: When the MaintenancePlanIdentifier attribute appears without a MaintenanceProcessIdentifier attribute, the reference is to all of the MaintenanceProcess instances in the MaintenancePlan. The interpretation is the same as if a MaintenancePlanReference instance had been used.

6.6.2.6 OrderInformationReference Class

The OrderInformationReference class provides a means to create a reference to an Order instance, and optionally to an OrderLine class instance within that Order instance.

6.6.2.6.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi-plicity	Description
OrderIdentifier	Identifier	1	The identifier of the Order class instance being referenced.
OrderLineIdentifier	Identifier	0 to 1	An identifier of an OrderLine class instance within the Order instance.

6.6.2.6.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>AbstractEntityReference</i> (subclass of <i>ReferencingEntity</i>)	DocumentIdentifier.

6.6.2.6.3 Constraints and Notes

None.

6.6.2.7 ProcessReference Class

The ProcessReference class provides a means to create a reference to a Process class instance.

6.6.2.7.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ProcessPlanIdentifier	Identifier	0 to 1	The identifier of the ProcessPlan class instance containing the Process instance being referenced.
ProcessIdentifier	Identifier	0 to 1	The identifier of a Process class instance within the ProcessPlan instance.

6.6.2.7.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>AbstractEntityReference</i> (subclass of <i>ReferencingEntity</i>)	DocumentIdentifier.

6.6.2.7.3 Constraints and Notes

Constraint: In valid instances of the *ProcessReference* class, at least one of the attributes *ProcessPlanIdentifier* and *ProcessIdentifier* shall be present.

Note: When the *ProcessPlanIdentifier* attribute appears without the *ProcessIdentifier* attribute, the interpretation is the reference is to all of the Process instances in the *ProcessPlan*. The interpretation is the same as if a *ProcessPlanReference* instance had been used.

6.6.2.8 ScheduleInformationReference Class

The ScheduleInformationReference class provides a means to create a reference to a Schedule instance, and optionally to a ScheduleItem class instance within that Schedule instance.

6.6.2.8.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ScheduleIdentifier	Identifier	1	The identifier of the Schedule class instance being referenced.
ScheduleItemIdentifier	Identifier	0 to 1	An identifier of a ScheduleItem class instance within the Schedule instance.

6.6.2.8.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>AbstractEntityReference</i> (subclass of <i>ReferencingEntity</i>)	DocumentIdentifier.

6.6.2.8.3 Constraints and Notes

None.

6.6.2.9 SkillReference Class

The SkillReference class provides a means to create a reference to a SkillDefinition class instance, and optionally to a SkillLevel class instance within that SkillDefinition instance.

6.6.2.9.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
SkillDefinitionIdentifier	Identifier	1	The identifier of the SkillDefinition class instance being referenced.
SkillLevelIdentifier	Identifier	0 to 1	An identifier of a SkillLevel class instance within the SkillDefinition instance.

6.6.2.9.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>AbstractEntityReference</i> (subclass of <i>ReferencingEntity</i>)	DocumentIdentifier.

6.6.2.9.3 Constraints and Notes

None.

6.7 Layout Package

The Layout package defines classes and relationships that facilitate the creation of manufacturing layout information. A manufacturing layout is a specification of the spatially-oriented characteristics and interrelationships for the logical and physical entities that are used to carry out production activities. Layout information can be used to provide visualizations of manufacturing facilities, or can be used as input to applications that programmatically analyze the interrelationships between the manufacturing entities contained in the layout.

6.7.1 Diagrams

6.7.1.1 LayoutElement and Related Classes

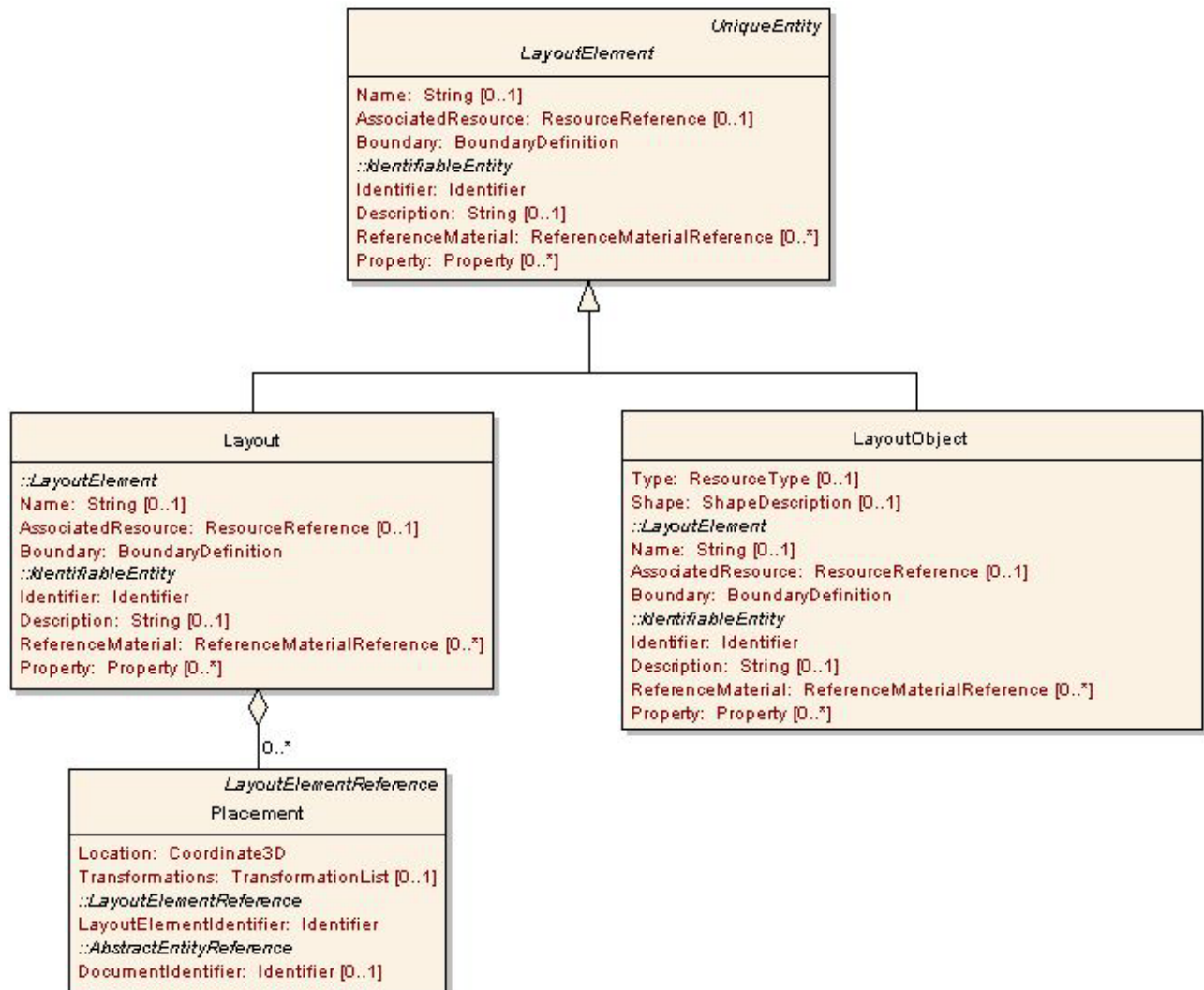


Figure 22: LayoutElement and Related Classes

The Layout class specifies information about a space in which production activities may take place, referred to as a manufacturing layout or a layout. This includes information about the boundaries of the space and a coordinate system to provide a frame of reference for objects within the space. The objects within the space

defined by a layout are referred to as layout elements. The position and orientation of layout elements within a layout are defined by instances of the Placement class.

Layout elements are either LayoutObject instances or other Layout instances. The *LayoutElement* class is an abstract generalization of the common properties of the Layout and LayoutObject classes.

The LayoutObject class represents a logical or physical entity that might be a part of a manufacturing layout. A LayoutObject's information includes boundary and coordinate system information about the space it occupies, and rudimentary information about the shape or image of the entity it represents.

The Placement class specifies the position and orientation of a *LayoutElement* within an associated Layout instance. It contains reference information for the *LayoutElement* to be placed, a point within the boundaries of the associated Layout where the *LayoutElement* should be placed, and a list of transformations that should be performed on the *LayoutElement* after placement.

6.7.1.2 ShapeDescription and Related Classes

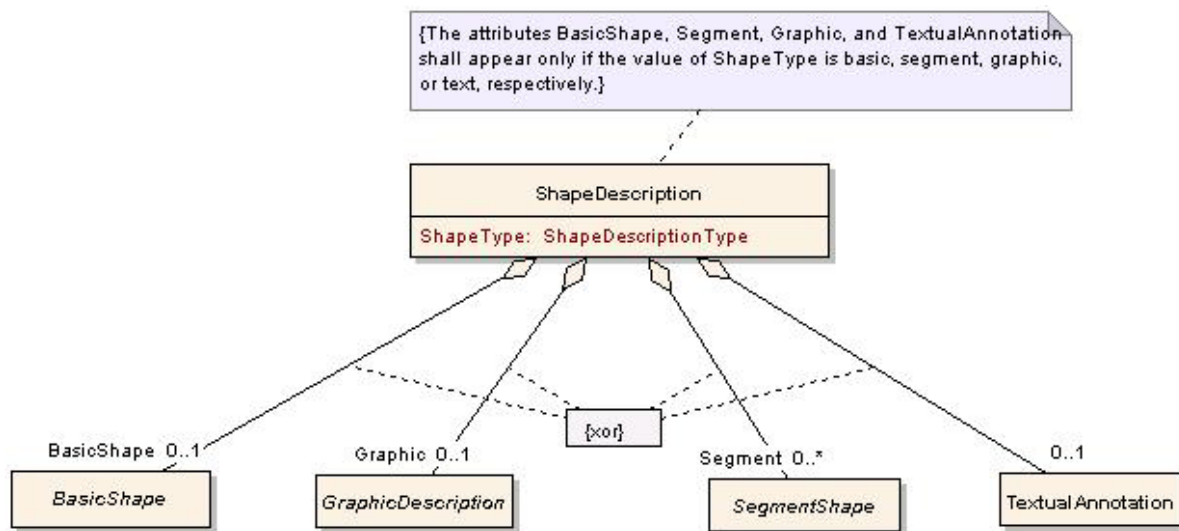


Figure 23: ShapeDescription and Related Classes

The *ShapeDescription* class describes the general form and appearance of an object that will be used to represent the shape of the manufacturing entity represented by a *LayoutObject*. The information describing the shape is one of four kinds. A *BasicShape* is a representation of a simple geometric shape such as a circle or polygon. A *Graphic* defines information about a file containing a 2D image or 3D computer graphics model. A *Segment* defines information about a component of an object that is composed of connected sections that operate together, such as a manufacturing conveyor. A *TextualAnnotation* defines a brief formatted text message that can be displayed as a part of a visualization of a layout. A *ShapeDescription* instance shall specify the kind of shape information that it contains and only that kind of shape information may be contained in that instance.

6.7.1.3 TransformationList and Related Classes

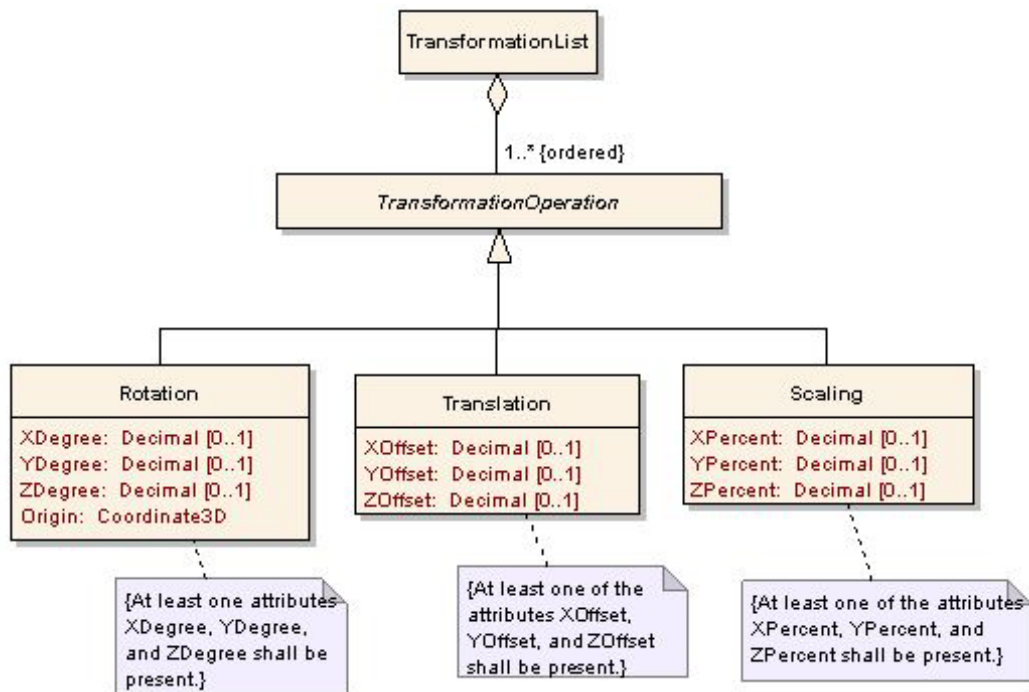


Figure 24: TransformationList and Related Classes

The *TransformationList* class represents an ordered list of modifications to the position or size of the object associated with a *LayoutElement* subclass instance. This list indicates how the object should be rotated, translated, and scaled after it is placed within a *Layout*.

A *TransformationOperation* class is an abstract generalization of the different transformations that may be performed on a *LayoutElement*.

The *Rotation* Class specifies, for each coordinate system axis, the magnitude for a rotation operation on the associated *LayoutElement*. Magnitude information for at least one axis shall be included. The point around which the *LayoutElement* should be rotated shall also be provided.

The *Translation* Class specifies, for each coordinate system axis, the magnitude for a translation operation on the associated *LayoutElement*. Magnitude information for at least one axis shall be included.

The *Scaling* Class specifies, for each coordinate system axis, the magnitude for a scaling operation on the associated *LayoutElement*. Magnitude information for at least one axis shall be included.

6.7.1.4 *GraphicDescription* and Related Classes

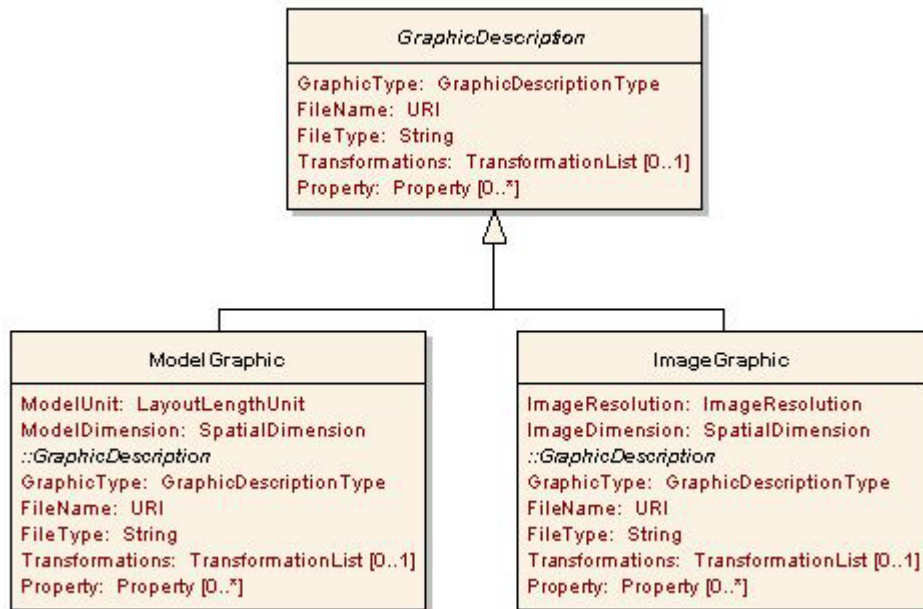


Figure 25: *GraphicDescription* and Related Classes

The *GraphicDescription* class is an abstract generalization of the common attributes of the *ModelGraphic* and *ImageGraphic* classes. It provides a means to specify information about a file containing model or image graphic data.

The *ModelGraphic* class defines unit and dimensional information about a file containing a machine-interpretable geometric representation of an object or objects. The *ImageGraphic* class defines resolution and dimensional information about a file containing a 2D image of some object or objects.

6.7.1.5 The *TextualAnnotation* Class

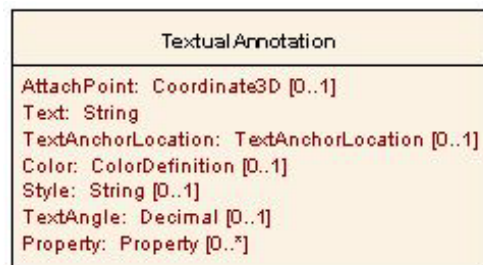


Figure 26: The *TextualAnnotation* Class

The *TextualAnnotation* class defines alignment, formatting, and style information for text information that is to be displayed within the boundaries of a *LayoutObject*.

6.7.1.6 *BasicShape* and Related Classes

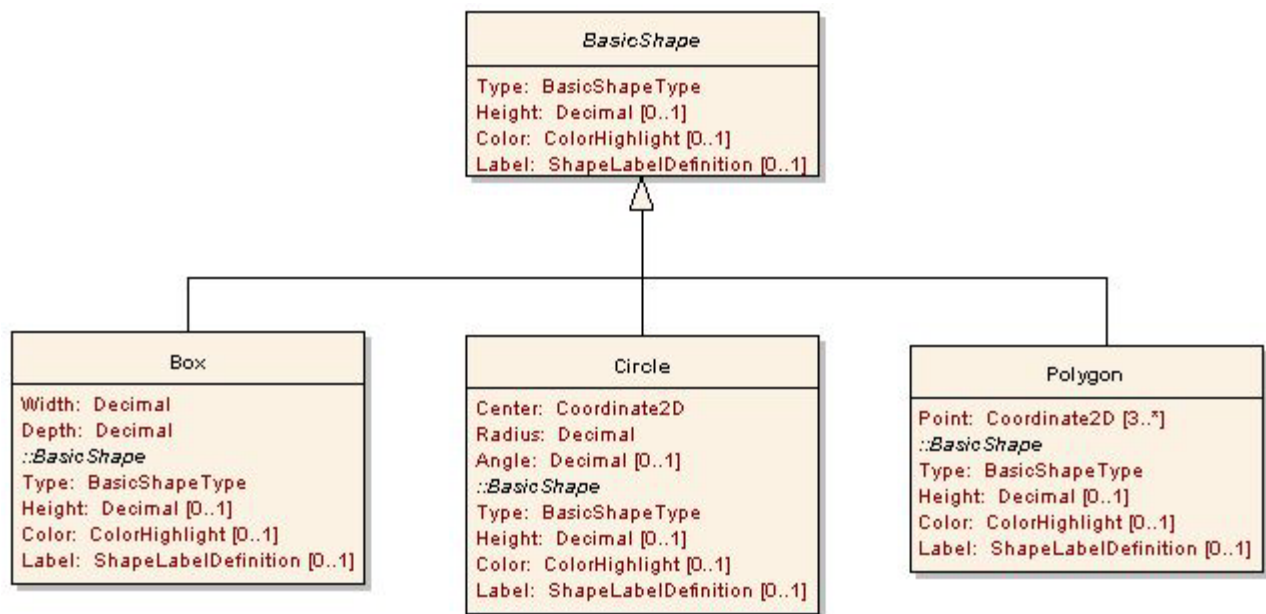


Figure 27: *BasicShape* and Related Classes

The *BasicShape* class is an abstract representation of the common attributes that can be used to define a simple shape that can be used as a visual representation of a *LayoutObject*. It defines information about the type, height, and color of the shape, and also about a label that can be displayed in visual representations of a *Layout*.

The *Box* class specifies information about a *BasicShape* that is in the form of a rectangle or a cuboid (3D rectangular box.)

The *Circle* class specifies information about a *BasicShape* that is in the form of a circle, a cylinder, or a “pie-wedge” shaped circular sector.

The *Polygon* class specifies information about a *BasicShape* that is in the form of a closed polygonal shape or a polygonal prism (3D polygonal shape.)

The information provided by these classes can be used to describe the shape of *LayoutObject*.

6.7.1.7 *SegmentShape* and Related Classes

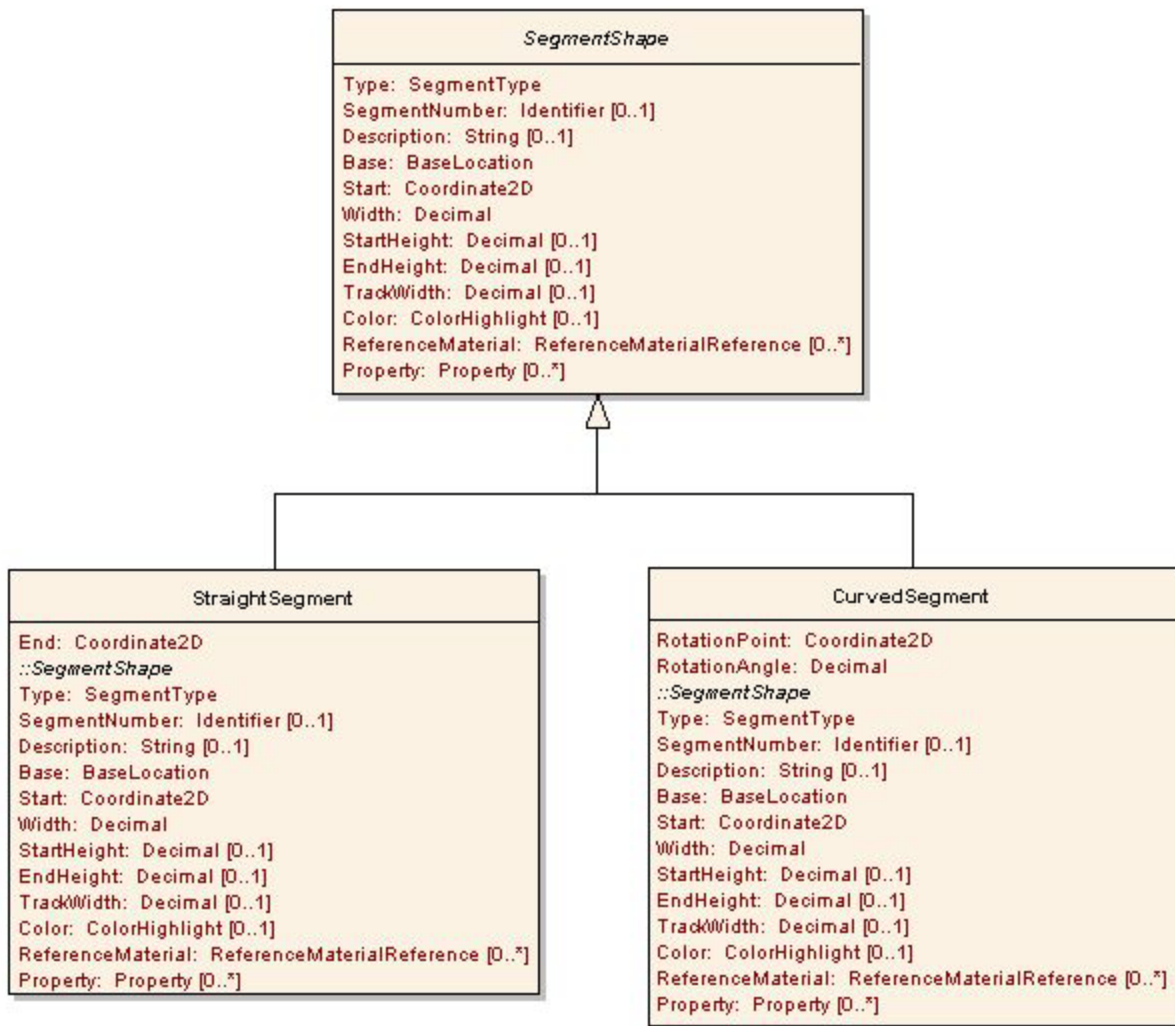


Figure 28: SegmentShape and Related Classes

The *SegmentShape* class is an abstract representation of the common information that is used to define a segment of a shape that is made up of interrelated sections. Some common manufacturing resources that have this form are conveyor systems. SegmentShapes may be of two types, curved or straight.

The *StraightSegment* class defines information for a segment that has the form of either a rectangle (2D) or a cuboid (3D).

The *CurvedSegment* class defines information for a segment that has the form of either a curved rectangle (2D) or a curved cuboid (3D).

The information provided by these classes can be used to describe the shape of a *LayoutObject*.

6.7.2 Classes

6.7.2.1 *BasicShape* Class

A *BasicShape* is an abstract representation of the common properties of the simple shapes that can be used to provide a basic description of the shape of a layout object.

6.7.2.1.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Type	BasicShapeType	1	The kind of basic shape that is used to represent the layout object, either a box, circle, or polygon.
Height	Decimal	0 to 1	The distance that the basic shape rises from the base of the boundary of the layout object.
Color	ColorHighlight	0 to 1	Color highlighting information that can be used to distinguish this layout object from others.
Label	ShapeLabelDefinition	0 to 1	A brief description of the layout object.

6.7.2.1.2 Superclasses & Inherited Attributes

None.

6.7.2.1.3 Constraints and Notes

Note: The base for all basic shapes is the floor of the layout object's boundary.

6.7.2.2 Box Class

A Box is a basic shape that represents either a rectangle (2D) or a cuboid (3D).

6.7.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Width	Decimal	1	The magnitude of the width of the box.
Depth	Decimal	1	The magnitude of the depth of the box.

6.7.2.2.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>BasicShape</i>	Type, Height, Color, Label.

6.7.2.2.3 Constraints and Notes

None.

6.7.2.3 Circle Class

A Circle is a *BasicShape* that represents a circle (2D), a cylinder (3D), or a “pie-wedge” shaped circular sector (2D or 3D).

6.7.2.3.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Center	Coordinate2D	1	The location of the center of the circle.
Radius	Decimal	1	The radius of the circle.
Angle	Decimal	0 to 1	The angle (in degrees) formed by two lines from the center to the endpoints of a circular sector.

6.7.2.3.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>BasicShape</i>	Type, Height, Color, Label.

6.7.2.3.3 Constraints and Notes

None.

6.7.2.4 CurvedSegment Class

A CurvedSegment is a *SegmentShape* that defines information for a segment that has the general form of a curved rectangle or a curved cuboid.

6.7.2.4.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
RotationPoint	Coordinate2D	1	The point around which the segment curves.

Attribute/Role Name	Type	Multi- plicity	Description
RotationAngle	Decimal	1	The angular distance in degrees that the curved segment rotates from its starting point.

6.7.2.4.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>SegmentShape</i>	Type, SegmentNumber, Description, Base, Start, Width, StartHeight, EndHeight, TrackWidth, Color, ReferenceMaterial, Property.

6.7.2.4.3 Constraints and Notes

None.

6.7.2.5 *GraphicDescription* Class

A *GraphicDescription* is an abstract representation of the common properties of a model graphic or image graphic.

6.7.2.5.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
GraphicType	GraphicDescriptionType	1	An indication of whether a graphic description is for a model or an image graphic.
FileName	URI	1	The name and location of the file containing the graphic data.
FileType	String	1	An indication of the type or format of graphic data contained in the associated file.
Transformations	TransformationList	0 to 1	An ordered list of translation, rotation, and scaling operations to be applied to a model or image graphic object to position it within the boundaries of its associated LayoutObject instance.

Attribute/Role Name	Type	Multi- plicity	Description
Property	Property	0 to *	Name and value information for a noteworthy characteristic of a graphic description. The allowable format and value space for a Property can be defined by creating a PropertyDescription and associating the Property with it.

6.7.2.5.2 Superclasses & Inherited Attributes

None.

6.7.2.5.3 Constraints and Notes

None.

6.7.2.6 ImageGraphic Class

An ImageGraphic is a 2D representation of an image or picture, made up of color and intensity data for each point (referred to as a pixel) in a rectangular grid.

6.7.2.6.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ImageResolution	ImageResolution	1	The length unit and pixels-per-unit associated with the image.
ImageDimension	SpatialDimension	1	The size of the rectangular grid containing the image data.

6.7.2.6.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>GraphicDescription</i>	GraphicType, FileName, FileType, Transformations, Property.

6.7.2.6.3 Constraints and Notes

None.

6.7.2.7 Layout Class

A Layout is a representation of the boundaries of a space in which manufacturing operations take place, and definitions for the position, orientation, and boundaries of manufacturing-related objects identified as existing within that space. These objects are defined by instances of the LayoutObject class and other Layout class instances.

6.7.2.7.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Placement	Placement	0 to *	The position, orientation, and scaling of a layout element within a layout.

6.7.2.7.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LayoutElement</i> (subclass of <i>UniqueEntity</i>)	Name, AssociatedResource, Boundary, Identifier, Description, ReferenceMaterial, Property.

6.7.2.7.3 Constraints and Notes

None.

6.7.2.8 LayoutElement Class

A *LayoutElement* is an abstract representation of the common properties of a Layout or LayoutObject.

6.7.2.8.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Name	String	0 to 1	The name of the <i>LayoutElement</i> .
AssociatedResource	ResourceReference	0 to 1	A reference to a resource for which some spatial aspect of that resource is, in some way, described by the other information in the accompanying <i>LayoutElement</i> subclass instance.
Boundary	BoundaryDefinition	1	The dimensions of the space occupied by the layout element.

6.7.2.8.2 Superclasses and Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.7.2.8.3 Constraints and Notes

None.

6.7.2.9 LayoutObject Class

A LayoutObject is a representation of the boundaries and general shape of a manufacturing-related object that may exist within the boundaries of a Layout.

6.7.2.9.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Type	ResourceType	0 to 1	The kind of manufacturing resource a layout object represents.
Shape	ShapeDescription	0 to 1	The general shape of a layout object.

6.7.2.9.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LayoutElement</i> (subclass of <i>UniqueEntity</i>)	Name, AssociatedResource, Boundary, Identifier, Description, ReferenceMaterial, Property.

6.7.2.9.3 Constraints and Notes

None.

6.7.2.10 ModelGraphic Class

A ModelGraphic is a 2D or 3D geometric representation of a real or conceptual object.

6.7.2.10.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ModelUnit	LayoutLengthUnit	1	The unit for distance associated with the object in the model.
ModelDimension	SpatialDimension	1	The boundaries of the object specified in the model.

6.7.2.10.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>GraphicDescription</i>	GraphicType, FileName, FileType, Transformations, Property.

6.7.2.10.3 Constraints and Notes

None.

6.7.2.11 Placement Class

A Placement specifies position, orientation, and scaling information for a *LayoutElement* within a Layout.

6.7.2.11.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Location	Coordinate3D	1	The layout element's position within the boundaries of its associated layout.
Transformations	TransformationList	0 to 1	An ordered list of translation, rotation, and scaling operations to be applied to a layout element after its placement within a Layout.

6.7.2.11.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
LayoutElementReference	LayoutElementIdentifier, DocumentIdentifier.

6.7.2.11.3 Constraints and Notes

None.

6.7.2.12 Polygon Class

A Polygon is a *BasicShape* containing an ordered list of points that represent either a closed polygon (2D) or a polygonal prism (3D).

6.7.2.12.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Point	Coordinate2D	3 to *	A vertex of the polygon.

6.7.2.12.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>BasicShape</i>	Type, Height, Color, Label.

6.7.2.12.3 Constraints and Notes

Note: All Polygons are closed. That is, for all Point attributes except the last in the list of Point attributes contained in a Polygon, a line from one Point to the next Point defines one side of the Polygon. In addition, a line from the last Point in the list to the first Point in the list also defines a side of the Polygon.

6.7.2.13 Rotation Class

A Rotation specifies how a *LayoutElement* should be rotated after placement in a Layout.

6.7.2.13.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
XDegree	Decimal	0 to 1	The angular distance in degrees that the associated layout element should be rotated with respect to the x-axis.
YDegree	Decimal	0 to 1	The angular distance in degrees that the associated layout element should be rotated with respect to the y-axis.
ZDegree	Decimal	0 to 1	The angular distance in degrees that the associated layout element should be rotated with respect to the z-axis.
Origin	Coordinate3D	1	The point around which the layout element is rotated.

6.7.2.13.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>TransformationOperation</i>	None.

6.7.2.13.3 Constraints and Notes

Constraint: At least one of the attributes XDegree, YDegree, and ZDegree shall appear in valid instances of Rotation.

6.7.2.14 Scaling Class

A Scaling is a *TransformationOperation* that specifies how a *LayoutElement* should be scaled after placement in a Layout.

6.7.2.14.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
XPercent	Decimal	0 to 1	The amount that the associated layout element should be scaled with respect to the x-axis.
YPercent	Decimal	0 to 1	The amount that the associated layout element should be scaled with respect to the y-axis.

Attribute/Role Name	Type	Multi- plicity	Description
ZPercent	Decimal	0 to 1	The amount that the associated layout element should be scaled with respect to the z-axis.

6.7.2.14.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>TransformationOperation</i>	None.

6.7.2.14.3 Constraints and Notes

Constraint: At least one of the attributes XPercent, YPercent, and ZPercent shall appear in valid instances of Scaling.

6.7.2.15 *SegmentShape* Class

A *SegmentShape* defines information about a segment of a shape that is made up of related sections.

6.7.2.15.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Type	SegmentType	1	Defines whether this section is straight or curved.
SegmentNumber	Identifier	0 to 1	An identifier that can be used to differentiate different segments in a layout object from each other.
Description	String	0 to 1	A description of this segment.
Base	BaseLocation	1	Defines whether the section is attached to the floor or the ceiling of the boundary of the layout object
Start	Coordinate2D	1	The point on the base where the segment starts.
Width	Decimal	1	The width of the segment.
StartHeight	Decimal	0 to 1	The height of the segment at the starting point of the segment.
EndHeight	Decimal	0 to 1	The height of the segment at the ending point of the segment.

Attribute/Role Name	Type	Multi- plicity	Description
TrackWidth	Decimal	0 to 1	A width of the region that runs down the center of a segment that defines the area over which objects on the can travel.
Color	ColorHighlight	0 to 1	Color highlighting information that can be used to distinguish this segment from others.
ReferenceMaterial	ReferenceMaterialReference	0 to *	A reference to information that is related to this segment shape.
Property	Property	0 to *	Name and value information for a noteworthy characteristic of a segment shape.

6.7.2.15.2 Superclasses & Inherited Attributes

None.

6.7.2.15.3 Constraints and Notes

None.

6.7.2.16 ShapeDescription Class

A ShapeDescription defines rudimentary visualization information for the object or objects that exist within the boundary of a layout object.

6.7.2.16.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ShapeType	ShapeDescriptionType	1	The kind of shape defined by this shape description.
BasicShape	<i>BasicShape</i>	0 to 1	A definition for a 2D or 3D simple geometric shape.
Graphic	<i>GraphicDescription</i>	0 to 1	Information about a file containing an image or model of an object.
Segment	<i>SegmentShape</i>	0 to *	Definition for a shape made up of related straight and/or curved cuboid sections.
TextualAnnotation	TextualAnnotation	0 to 1	Descriptive text for a layout.

6.7.2.16.2 Superclasses & Inherited Attributes

None.

6.7.2.16.3 Constraints and Notes

- Constraint:** The attributes BasicShape, Graphic, Segment, and TextualAnnotation shall only be used as attributes of a ShapeDescription instance if the value of the ShapeType attribute is “basic”, “graphic”, “segment”, or “text”, respectively.
- Note:** Irrespective of the coordinate system used, width indicates distance along the x-axis, depth indicates distance along the y-axis, and height indicates distance along the z-axis.
- Note:** The base of a LayoutObject’s boundary determines how shapes are initially oriented within that boundary. The base is either the “floor” or “ceiling” of the LayoutObject’s boundary. Floor denotes the part of the boundary perpendicular with the xy-plane and through the minimum z value for that boundary. Ceiling denotes the part of the boundary perpendicular with the xy-plane and through the maximum z value for that boundary.
- Note:** The minimum and maximum z values for a boundary depend on the coordinate system associated with that boundary. For upper left based coordinate systems, the minimum z value is 0 and maximum z value is the height defined for the boundary. For center based coordinate systems, the minimum z value is -height/2 and maximum z value is +height/2 where height is the height defined for the boundary.
- Note:** A 3D shape in a CMSD layout can be interpreted as being formed by “drawing” 2 dimensional aspects of the shape on the base of the boundary, and then raising/extruding the 2 dimensional shape orthogonally from the base for the distance defined by the shape’s height. For boundaries whose base is defined as “ceiling”, this raising is done in the negative z direction.

6.7.2.17 StraightSegment Class

A StraightSegment is a *SegmentShape* that defines information for a segment that has the general form of a rectangle or a cuboid.

6.7.2.17.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
End	Coordinate2D	1	The point on the base where the segment ends.

6.7.2.17.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>SegmentShape</i>	Type, SegmentNumber, Description, Base, Start, Width, StartHeight, EndHeight, TrackWidth, Color, ReferenceMaterial, Property.

6.7.2.17.3 Constraints and Notes

None.

6.7.2.18 TextualAnnotation Class

A TextualAnnotation defines information about descriptive text that can be added to a LayoutObject to enhance the understanding of a Layout.

6.7.2.18.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
AttachPoint	Coordinate3D	0 to 1	The point in the layout object's boundary where the anchor of the text is placed.
Text	String	1	The text of the textual annotation.
TextAnchorLocation	TextAnchorLocation	0 to 1	Specifies whether the text anchor is located at the upper left of the text or if it is located in the center of the text.
Color	ColorDefinition	0 to 1	The color to be used for the text.
Style	String	0 to 1	A list of CSS2 compliant text and font property declarations.
TextAngle	Decimal	0 to 1	Indicates the text should be rotated around the anchor for the specified angle in degrees.
Property	Property	0 to *	Name and value information for a noteworthy characteristic of a text annotation.

6.7.2.18.2 Superclasses & Inherited Attributes

None.

6.7.2.18.3 Constraints and Notes

- Note: The base for all TextualAnnotations is the floor of its LayoutObject's boundary.
- Note: TextualAnnotations are 2 dimensional. If the AttachPoint contains a value for Z, the TextualAnnotation should be considered to be defined on a plane parallel the xy-plane and through the Z value. TextualAnnotations are not "raised" from the base to form 3D shapes as other kinds of shapes are.
- Note: The anchor of a TextualAnnotation instance defines a relative point to which the text defined for that instance is oriented. The location of the anchor is either the upper left corner of the text or the center (both horizontal and vertical) of the text.
- Note: If the TextAnchorLocation attribute is not present, a TextualAnnotation's anchor is located at the upper left corner of the text (if the LayoutObject's coordinate system is upper left based) or at the center of the text (if the LayoutObject's coordinate system is center based).
- Note: The Style element provides formatting information about how TextualAnnotation information might be visualized. This information is intended to provide hints/suggestions/guidelines for effective visualization, but should not be viewed as requirements.

Note: The information in the Style attribute should adhere to that which is defined by the World Wide Web Consortium's (W3C) Cascading Style Sheet 2 recommendation (CSS2). While this standard was initially intended to describe formatting information for browsers, CSS2 style information is in wide use in a wide number of applications.

Note: The Style string should contain a list of CSS2 property value pairs, where the CSS2 property and its associated value are separated by a colon (":") and where pairs are separated by semicolons (;). The properties defined in the CSS2 standard for font and text formatting are appropriate for use in a TextualAnnotation Style string, but other properties may also be useful. See the CSS2 standard for the list of properties and appropriate property values.

6.7.2.19 TransformationList Class

The TransformationList class represents an ordered list of modifications to the position or size of the object associated with a LayoutElement subclass instance. This list indicates how the object should be rotated, translated, and scaled after it is placed within a Layout.

6.7.2.19.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Name of a <i>TransformationOperation</i> subclass	<i>TransformationOperation</i> subclass	1 to *	Information defining a change to the position, orientation, or scale of an associated layout or layout object.

6.7.2.19.2 Superclasses & Inherited Attributes

None.

6.7.2.19.3 Constraints and Notes

Note: Since *TransformationsOperation* is an abstract class, it can not appear directly as an attribute of a TransformationList instance, but any concrete subclass of *TransformationsOperation* is allowed to appear.

6.7.2.20 TransformationOperation Class

A *TransformationOperation* is an abstract generalization of information defining a change to the position, orientation, or scale of an associated Layout or LayoutObject.

6.7.2.20.1 Defined Attributes and Association Roles

None.

6.7.2.20.2 Superclasses & Inherited Attributes

None.

6.7.2.20.3 Constraints and Notes

None.

6.7.2.21 Translation Class

The Translation class specifies how a LayoutElement should be offset along the x-, y-, or z-axis after placement in a Layout.

6.7.2.21.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
XOffset	Decimal	0 to 1	The distance that the associated layout element should be translated along the x-axis.
YOffset	Decimal	0 to 1	The distance that the associated layout element should be translated along the y-axis.
ZOffset	Decimal	0 to 1	The distance that the associated layout element should be translated along the z-axis.

6.7.2.21.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>TransformationOperation</i>	None.

6.7.2.21.3 Constraints and Notes

Constraint: At least one of the attributes XOffset, YOffset, and ZOffset shall appear in valid instances of Translation.

6.8 Metadata Package

The Metadata package contains classes and relationships that can be used to define the allowable content of a Property attribute that occurs as a part of a CMSD entity.

In CMSD, Property attributes provide a means to enhance/refine/extend the definition of a CMSD entity with additional information about that entity's traits, features, or characteristics. This information is referred to as a "property" of the entity. The classes in the Metadata package can be used to define name, content, format, and applicable unit information for Property attributes. Taken together, Property attributes and the metadata information that describes them provide for the flexible extension of CMSD entities while also providing a means to specify the intent of those extensions within CMSD.

6.8.1 Diagrams

6.8.1.1 PropertyDescription and Related Classes

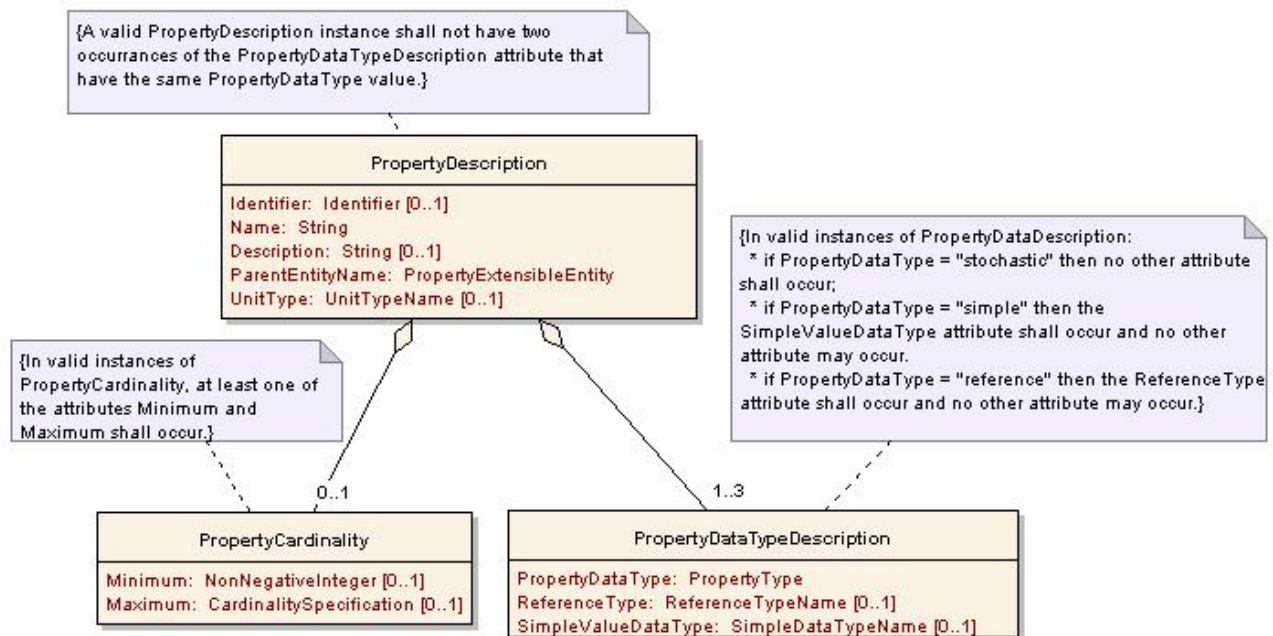


Figure 29: PropertyDescription and Related Classes

The PropertyDescription class defines information about the features of a Property attribute that can be used to define a "property" for a CMSD entity. This information includes the name of the CMSD entity in which the Property attribute appears (the parent entity), the name of the "property" being defined by the Property attribute, the allowable number of occurrences of that property within a parent entity, and the format of information that can be associated with that property.

The PropertyCardinality class defines the minimum and maximum number of times that a property may occur within its associated parent entity instance.

The PropertyDataTypeDescription class defines information specifying the type and format of allowable content for a property.

6.8.2 Classes

6.8.2.1 PropertyCardinality Class

The PropertyCardinality class defines information about the minimum and maximum allowable occurrences of a property within its associated parent entity instance.

6.8.2.1.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Minimum	NonNegativeInteger	0 to 1	The minimum number of occurrences of a property within its associated parent entity.
Maximum	CardinalitySpecification	0 to 1	The maximum number of occurrences of a property within its associated parent entity.

6.8.2.1.2 Superclasses & Inherited Attributes

None.

6.8.2.1.3 Constraints and Notes

Constraint: In a Valid PropertyCardinality class instance, at least one of the attributes Minimum and Maximum shall appear.

Note: If the Minimum attribute does not appear, 0 is the minimum number of occurrences that should be used for the associated property.

Note: If the Maximum attribute does not appear, 1 is the maximum number of occurrences that should be used for the associated property.

6.8.2.2 PropertyDataTypeDescription Class

The PropertyDataTypeDescription class defines information specifying the type and format of allowable content for a property.

6.8.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PropertyDataType	PropertyType	1	The information associated with a property that represents a simple value, a reference to CMSD entity instance, or a definition of a statistical distribution.
ReferenceType	ReferenceTypeName	0 to 1	The name of the CMSD referencing entity subclass that can be associated with a Property attribute.

Attribute/Role Name	Type	Multi- plicity	Description
SimpleValueType	SimpleDataTypeName	0 to 1	The format for simple data that can be associated with a Property attribute.

6.8.2.2.2 Superclasses & Inherited Attributes

None.

6.8.2.2.3 Constraints and Notes

Constraint: In valid PropertyDataTypeDescription class instances, the value of the PropertyDataType attribute determines whether the SimpleValueType or ReferenceType attribute shall appear.

- If the value of the PropertyDataType attribute is “stochastic” then no other attribute shall appear.
- If the value of the PropertyDataType attribute is “simple” then the SimpleValueType attribute shall appear and the ReferenceType attribute shall not appear.
- If the value of the PropertyDataType attribute is “reference” then the ReferenceType attribute shall appear and the SimpleValueType attribute shall not appear.

6.8.2.3 PropertyDescription Class

The PropertyDescription class defines information specifying the type and format of allowable content for a property.

6.8.2.3.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Identifier	Identifier	0 to 1	A sequence of characters allowing instances of this class to be differentiated from each other.
Name	String	1	The name of the property being described.
Description	String	0 to 1	Information describing what this property is meant to represent.
ParentEntityName	PropertyExtensibleEntity	1	The name of the CMSD entity in which this property may appear.
UnitType	UnitTypeName	0 to 1	The name of the unit associated with this property.
PropertyCardinality	PropertyCardinality	0 to 1	The minimum and/or maximum number of occurrences of the associated property allowed for the associated parent entity.

Attribute/Role Name	Type	Multiplicity	Description
PropertyDataTypeDescription	PropertyDataTypeDescription	1 to 3	A specification of the kind and format of data that may be associated with a property.

6.8.2.3.2 Superclasses & Inherited Attributes

None.

6.8.2.3.3 Constraints and Notes

Constraint: In valid PropertyDescription class instances with more than one PropertyDataTypeDescription attribute, the PropertyDataTypeDescription attributes shall have different values for their PropertyDataType attributes.

6.9 Part Information Package

The Part Information package defines classes and relationships for describing the raw materials, work-in-progress components, and finished products that are the inputs to and outputs of manufacturing processes. In addition, information about the component structure of parts and about the kinds and amounts of parts either completed or available to be used in production activities can also be defined.

6.9.1 Diagrams

6.9.1.1 The Part Class and PartType Class

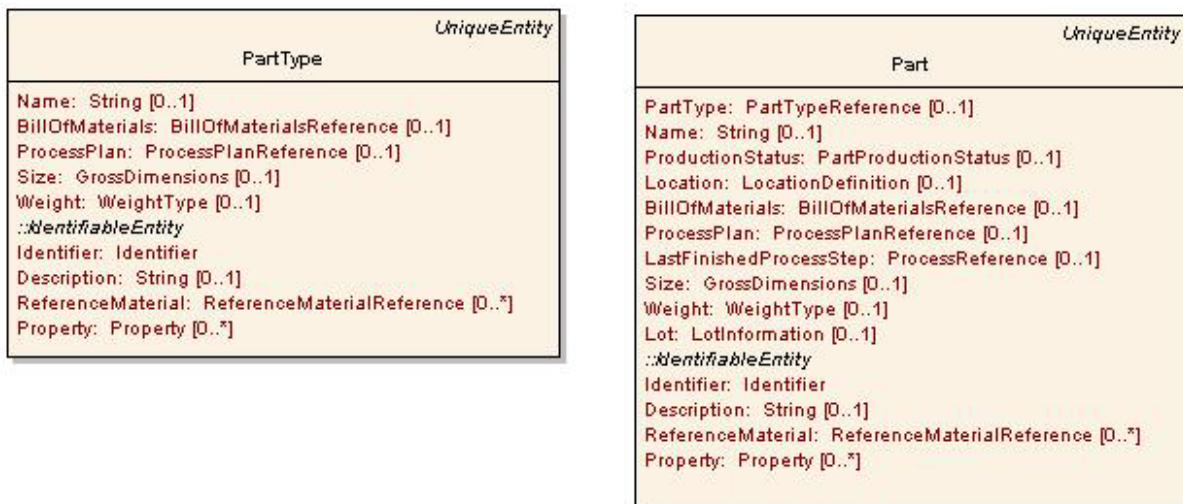


Figure 30: The Part Class and PartType Class

The Part class defines information about a specific instance of a part. A part is a raw material, work-in-process component, or finished product that was produced by or that can be used in production activities. Many different kinds of information can be specified for a part, such as, information about the production status of a part, the named category of parts that a specific part belongs to, the sub-component parts used to create this part, the process that can be used to create the part, and some basic characteristics of this part.

The PartType class defines information about a part type, which is a named category for parts with similar characteristics. The information that can be specified for a part type includes the basic characteristics associated with a part of this part type, the part types of the sub-components of parts of this part type, and the processes that can be used to create parts of this part type.

Additional information about the characteristics of a part or part type can be specified through the use of Property attributes in the associated Part class or PartType class instance.

6.9.1.2 BillOfMaterials and Related Classes

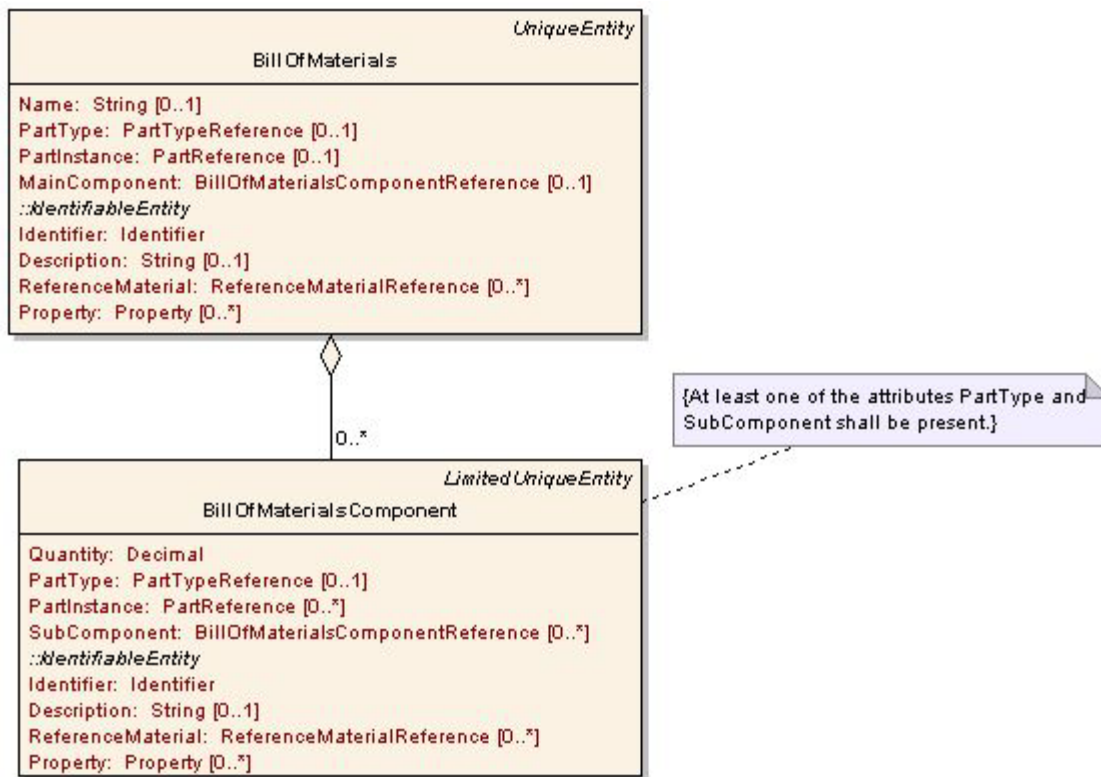


Figure 31: BillOfMaterials and Related Classes

The **BillOfMaterials** class defines information about the structure and relationships of the sub-component Parts and PartTypes that are used to create a Part.

The **BillOfMaterialsComponent** class defines information about a specific sub-component of a Part or PartType, and can also specify the hierarchy of a Part's sub-components through referencing the **BillOfMaterials** definitions for each this component's sub-components.

6.9.1.3 InventoryItem and Related Classes

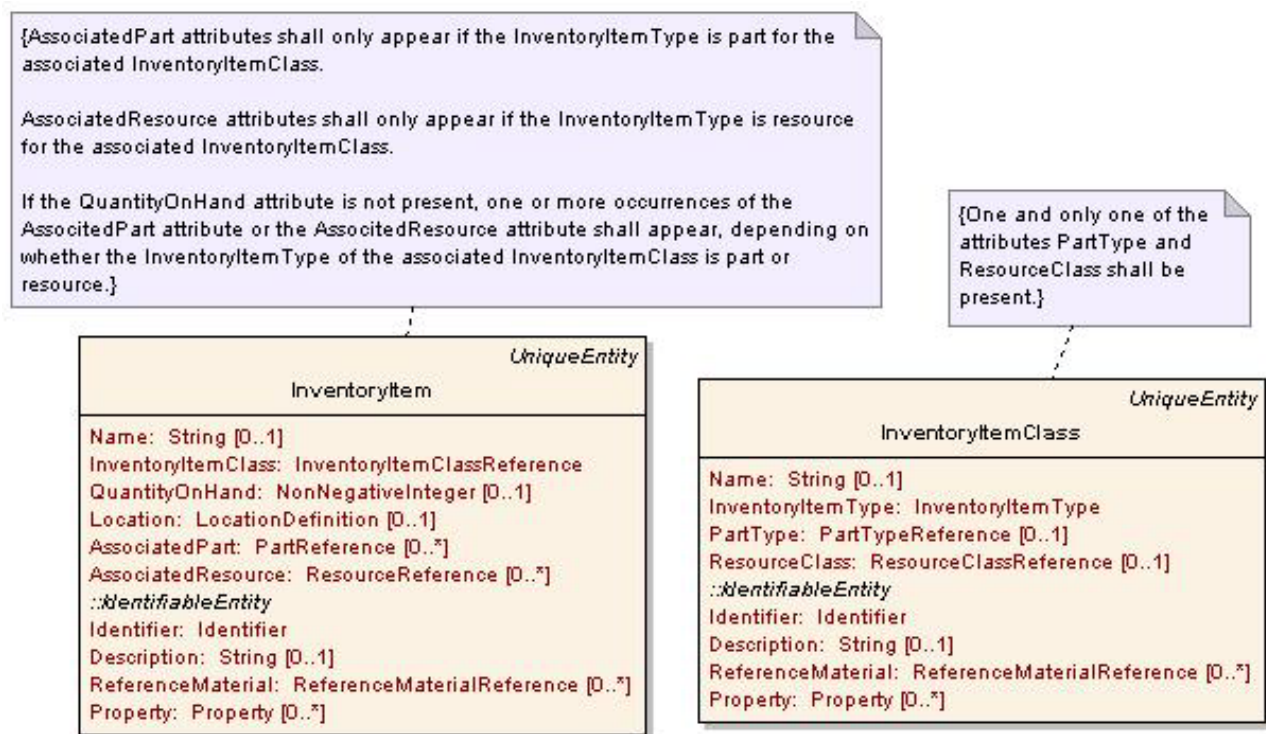


Figure 32: InventoryItem and Related Classes

The InventoryItemClass class provides a means to define a named category for a kind of inventory item, to specify whether that item is Part or Resource, and to specify information about the characteristics shared by all instances of that class.

The InventoryItem class defines information about the Part or Resource instances that are inventory items and that belong to a specific InventoryItemClass. It provides a means to specify how many of a given InventoryItem are available for production, to indicate the location of those items, and to describe the distinctive characteristics of a given InventoryItem instance that differentiates it from other InventoryItem instances of the same InventoryItemClass.

6.9.2 Classes

6.9.2.1 BillOfMaterials Class

The BillOfMaterials class defines information about the relationships between a part or part type and the sub-components that make up that part or part type.

6.9.2.1.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Name	String	0 to 1	The name of this bill of materials.

Attribute/Role Name	Type	Multi- plicity	Description
PartType	PartTypeReference	0 to 1	The type of part whose sub-component structure is defined by this bill of materials.
PartInstance	PartReference	0 to 1	A specific part instance whose sub-component structure is defined by this bill of materials
MainComponent	BillOfMaterialsComponentReference	0 to 1	The top component in the component hierarchy for the associated part type.
BillOfMaterialsComponent	BillOfMaterialsComponent	0 to *	Information describing the type and quantity of a sub-component of the Part or part type associated with this bill of materials.

6.9.2.1.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.9.2.1.3 Constraints and Notes

None.

6.9.2.2 BillOfMaterialsComponent Class

The BillOfMaterialsComponent class defines information about a sub-component of the Part or PartType whose structure is defined by a BillOfMaterials instance.

6.9.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Quantity	Decimal	1	The number of part sub-components this bill of materials component represents.
PartType	PartTypeReference	0 to 1	The type of part this bill of materials component represents.
PartInstance	PartReference	0 to 1	The specific part instance that this bill of materials component represents.

Attribute/Role Name	Type	Multi- plicity	Description
SubComponent	BillOfMaterialsComponent Reference	0 to *	A reference to a BillOfMaterials or BillOfMaterialsComponent instance that defines the substructure this BillOfMaterialsComponent.

6.9.2.2.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.9.2.2.3 Constraints and Notes

Constraint: Valid instances of the BillOfMaterialsComponent class shall contain at least one of the attributes PartType and SubComponent.

6.9.2.3 InventoryItem Class

The InventoryItem class defines information about Part or Resource instances whose state and availability to be used in production activities are tracked.

6.9.2.3.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Name	String	0 to 1	The name of the inventory item.
InventoryItemClass	InventoryItemClassReference	1	The class defined for the inventory item.
QuantityOnHand	NonNegativeInteger	0 to 1	The number of parts or resources associated with this inventory item.
Location	LocationDefinition	0 to 1	The place where the inventory item currently resides.
AssociatedPart	PartReference	0 to *	The specific parts that this inventory item represents.
AssociatedResource	ResourceReference	0 to *	The specific resources that this inventory item represents.

6.9.2.3.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.9.2.3.3 Constraints and Notes

Constraint: If the AssociatedPart attribute appears, the value of the associated InventoryItemClass.InventoryItemType attribute shall be “part”.

Constraint: If the AssociatedResource attribute appears, the value of the associated InventoryItemClass.InventoryItemType shall be “resource”.

Constraint: If the QuantityOnHand attribute is not present, one or more occurrences of the AssociatedPart attribute or the AssociatedResource attribute shall appear, depending on whether the InventoryItemType of the associated InventoryItemClass is “part” or “resource”.

6.9.2.4 InventoryItemClass Class

The InventoryItemClass class defines information about a specific category of parts or resources that are inventory items.

6.9.2.4.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Name	String	0 to 1	The name of the inventory item class.
InventoryItemType	InventoryItemType	1	Defines whether this class is for part or resource inventory items.
PartType	PartTypeReference	0 to 1	A reference to the part type of the parts that belong to this inventory item class.
ResourceClass	ResourceClassReference	0 to 1	A reference to the resource class of resources that belong to this inventory item class.

6.9.2.4.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.9.2.4.3 Constraints and Notes

Constraint: The PartType attribute shall only be present if the value of the InventoryItemType attribute is “part”, and the ResourceClass attribute shall only be present if the value of the associated InventoryItemType attribute is “resource”.

6.9.2.5 Part Class

The Part class defines information about a specific instance of a raw material, work-in-process component, or finished product that can be used in or produced by the manufacturing process.

6.9.2.5.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
PartType	PartTypeReference	0 to 1	The name of the kind of part being defined here.
Name	String	0 to 1	The name of the part.
ProductionStatus	PartProductionStatus	0 to 1	The state of the part with respect to production.
Location	LocationDefinition	0 to 1	The place where the part currently resides.
BillOfMaterials	BillOfMaterialsReference	0 to 1	A reference to the bill of materials defining the component structure of the part.
ProcessPlan	ProcessPlanReference	0 to 1	A reference to the process plan defining the process for creating the part.
LastFinishedProcessStep	ProcessReference	0 to 1	An indication of how much of the process to create the part has been completed.
Size	GrossDimensions	0 to 1	The approximate size of this part.
Weight	WeightType	0 to 1	The weight and weight unit for the part.
Lot	LotInformation	0 to 1	Information about a group of parts that were manufactured or acquired together.

6.9.2.5.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.9.2.5.3 Constraints and Notes

None.

6.9.2.6 PartType Class

The PartType class defines information about a distinct category of Part instances that have common characteristics.

6.9.2.6.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Name	String	0 to 1	The name of the part type.
BillOfMaterials	BillOfMaterialsReference	0 to 1	A reference to a bill of materials that defines the substructure of parts of this part type.
ProcessPlan	ProcessPlanReference	0 to 1	A reference to the process plan defining the process for creating the parts of this part type.
Size	GrossDimensions	0 to 1	The approximate size for parts of this part type.
Weight	WeightType	0 to 1	The weight and weight unit for parts of this part type.

6.9.2.6.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.9.2.6.3 Constraints and Notes

None.

6.10 Production Operations Package

The Production Operations package defines classes and relationships describing customer requests for products, production requests to produce those products, and the effort needed to produce those products.

6.10.1 Diagrams

6.10.1.1 Order and Job Related Classes

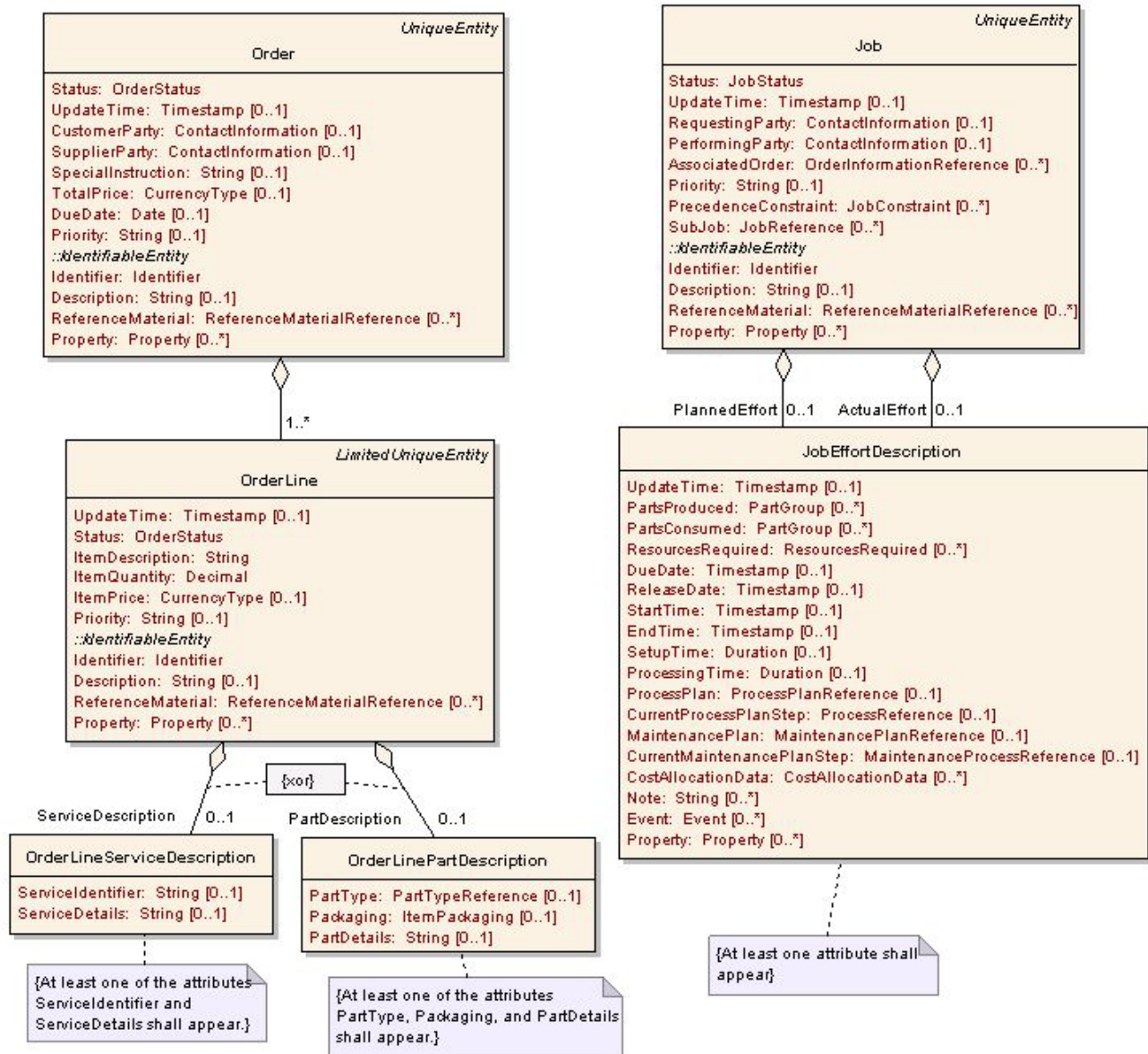


Figure 33: Order and Job Related Classes

The Order class defines a customer request for products or services, including information about the products or services ordered, when the product or service will be supplied, the price the customer will need to pay, and the processing status of the order.

The OrderLine class defines information about a product or service, referred to as an item, requested in an order. It includes information about the quantity ordered of the item, details about the product or service ordered, and the processing status for this item.

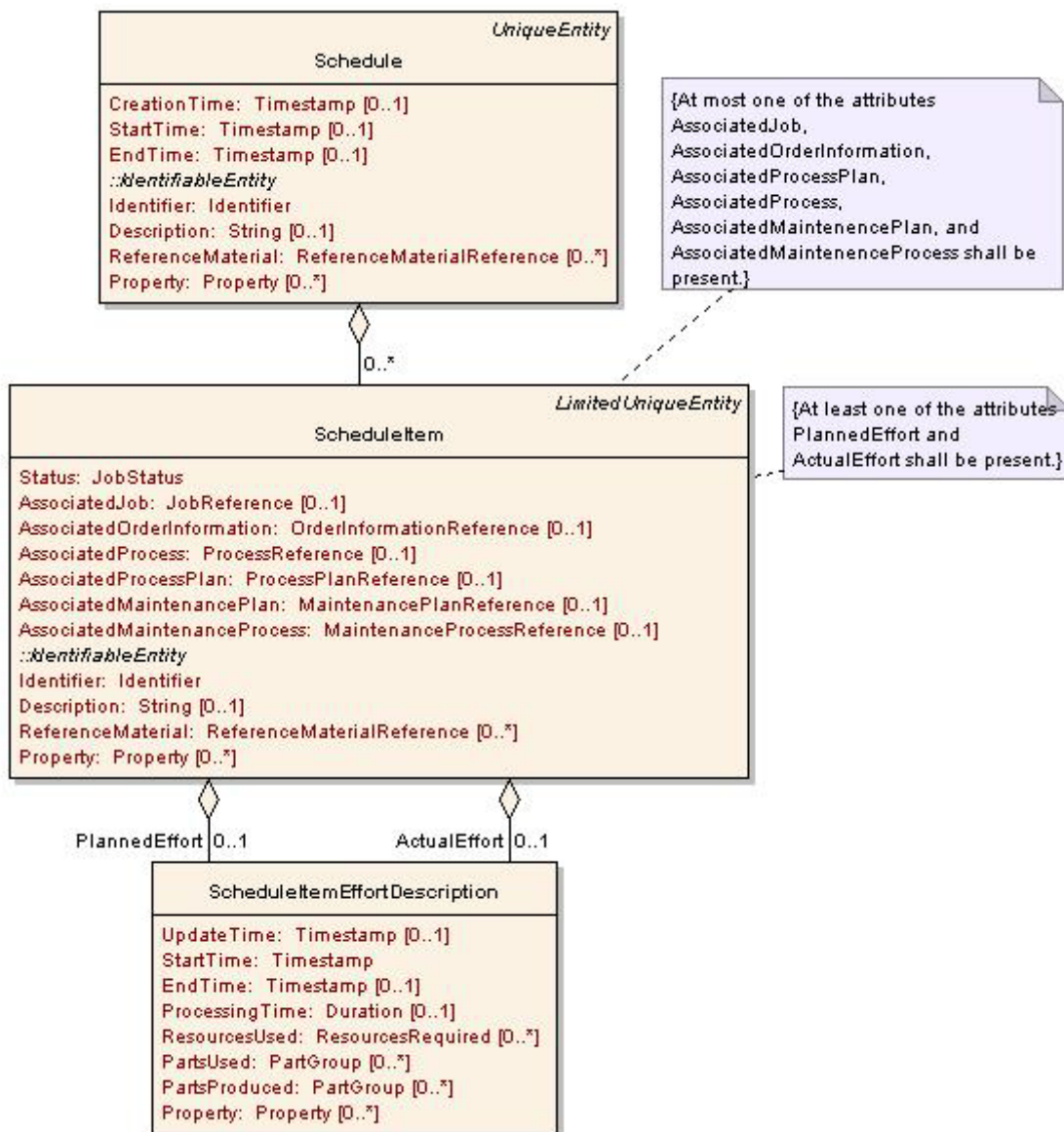
The OrderLineServiceDescription class provides a means to specify additional information about a service that was ordered. The information includes identifying information about the service and information about how the service is to be performed.

The OrderLinePartDescription class provides a means to specify additional information about a part that was ordered. The information includes the type of part that was ordered, how that part is packaged, and other information about the part.

The Job class defines information about a request to perform a production-related activity. A job may be a request to transform the state of parts and raw materials from a less finished form to a more finished form, requests to transfer parts from one location to another location, or request to perform maintenance activities on resources. It includes information about who requested the production activity, how this job is related to other jobs, and details about the effort needed to perform the job.

The JobEffortDescription class defines information about how the production activities associated with are to be carried out. It includes information about when the activities are expected to start and complete, how much actual production time will be used, what manufacturing resources will be needed, what parts will be produced or consumed, what predefined sequence of operations should be used to perform this job, and what costs will be incurred from performing the job.

6.10.1.2 The Schedule Class and Related Classes



The Schedule class defines information about the planned or actual starting and ending times for manufacturing-related activities that are planned to occur during a specified time period.

The ScheduleItem class defines information about an activity that is a part of a schedule. The activity may be associated with the processing of a job or order, or it may be related to the execution of a maintenance activity. The schedule item includes information about the completion status of the activity and how much time and how many resources were expended to carry out the activity.

The ScheduleItemEffortDescription class defines information about the time and resources expended to carry out a scheduled manufacturing-related activity. It includes information defining the time period during which the activity should take place, the resources used during the activity, and the parts consumed or produced during the activity.

6.10.2 Classes

6.10.2.1 Job Class

The Job class defines information about a request to perform a production activity.

6.10.2.1.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
Status	JobStatus	1	An indicator of the progress made towards completing processing of the request.
UpdateTime	Timestamp	0 to 1	The time of the last update to this job's information.
RequestingParty	ContactInformation	0 to 1	The person, organization, and/or business function requesting the production activity.
PerformingParty	ContactInformation	0 to 1	The person, organization, and/or business function responsible for carrying out the request.
AssociatedOrder	OrderInformationReference	0 to *	An order that is in some way related to this request.
Priority	String	0 to 1	An indication of the relative importance of this job in comparison to other jobs.
PrecedenceConstraint	JobConstraint	0 to *	A restriction on when a job may start or complete based on when a related job starts or completes.
SubJob	JobReference	0 to *	A group of related jobs that define/refine at a greater level of detail the processing specified for this job.
PlannedEffort	JobEffortDescription	0 to 1	Information describing the timing, amount, and kind of manufacturing activities that are planned to be carried out to complete the job.
ActualEffort	JobEffortDescription	0 to 1	Information describing the timing, amount, and kind of manufacturing activities that were carried out to complete the job.

6.10.2.1.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.10.2.1.3 Constraints and Notes

None.

6.10.2.2 JobEffortDescription Class

The JobEffortDescription class defines information describing the time, resource, material, and process related aspects of the manufacturing activities that are needed to complete a job.

6.10.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
UpdateTime	Timestamp	0 to 1	The time of the last of this information.
PartsProduced	PartGroup	0 to *	The parts that will be created by this job.
PartsConsumed	PartGroup	0 to *	The parts that are used up or that become subcomponents of other parts as a result of performing this job.
ResourcesRequired	ResourcesRequired	0 to *	The kind and quantity of resources needed to perform this job.
DueDate	Timestamp	0 to 1	The date the job should be completed on.
ReleaseDate	Timestamp	0 to 1	The earliest date that work can start for this job.
StartTime	Timestamp	0 to 1	The time when the job is to begin.
EndTime	Timestamp	0 to 1	The time when the job should be done.
SetupTime	Duration	0 to 1	The time it takes to assemble and configure resources to perform the job.
ProcessingTime	Duration	0 to 1	The amount of time between the starting and ending times that resources are actually performing manufacturing activities related to this job.

Attribute/Role Name	Type	Multi- plicity	Description
ProcessPlan	ProcessPlanReference	0 to 1	The detailed description of the manufacturing operations that are to be performed to carry out the job.
CurrentProcessPlan Step	ProcessReference	0 to 1	The current manufacturing operation being performed to complete the job.
MaintenancePlan	MaintenancePlanReference	0 to 1	The detailed description of the maintenance operations that are to be performed to carry out the job.
CurrentMaintenance PlanStep	MaintenanceProcessReference	0 to 1	The current maintenance operation being performed to complete the job.
CostAllocationData	CostAllocationData	0 to *	Information describing, categorizing, and estimating the amount of costs that may be incurred as a result of performing this job.
Note	String	0 to *	A comment or observation related to carrying out the manufacturing activities associated with the job.
Event	Event	0 to *	The occurrence or anticipated occurrence of some noteworthy situation or condition that happens between the starting and ending time of the job.
Property	Property	0 to *	Name and value information describing a noteworthy characteristic of a job.

6.10.2.2.2 Superclasses & Inherited Attributes

None.

6.10.2.2.3 Constraints and Notes

Constraint: In a valid JobEffortDescription instance, at least one of its attributes shall appear.

6.10.2.3 Order Class

The Order class defines information about a customer's request for products or services.

6.10.2.3.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Status	OrderStatus	1	An indicator of the progress made towards completing the order.

Attribute/Role Name	Type	Multiplicity	Description
UpdateTime	Timestamp	0 to 1	The time of the last update to this order's information.
CustomerParty	ContactInformation	0 to 1	The person, organization, and/or business function that requested the products or services defined in the order.
SupplierParty	ContactInformation	0 to 1	The person, organization, and/or business function responsible for providing the products or services associated with the order to the customer.
SpecialInstruction	String	0 to 1	Information describing an aspect of an order that is atypical, unusual, or unique, and that may affect how processing of the order should be carried out.
TotalPrice	CurrencyType	0 to 1	The price for all of the products and services defined in the order.
DueDate	Date	0 to 1	The date on which the order should be completed.
Priority	String	0 to 1	An indication of the relative importance of this order in comparison to other orders.
OrderLine	OrderLine	1 to *	A detailed description of a product or service that is to be provided to a customer.

6.10.2.3.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.10.2.3.3 Constraints and Notes

None.

6.10.2.4 OrderLine Class

The OrderLine class defines information about a specific product or service in a customer's request for products or services.

6.10.2.4.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
UpdateTime	Timestamp	0 to 1	The time of the last update to this order line's information.
Status	OrderStatus	1	An indicator of the progress made towards completing processing of this portion of the order.
ItemDescription	String	1	A general description of the product or service ordered.
ItemQuantity	Decimal	1	The number of items requested.
ItemPrice	CurrencyType	0 to 1	The price for this item.
ServiceDescription	OrderLineServiceDescription	0 to 1	Information about a service that was ordered.
PartDescription	OrderLinePartDescription	0 to 1	Information about a part that was ordered.
Priority	String	0 to 1	An indication of the relative importance of this order in comparison to other orders.

6.10.2.4.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.10.2.4.3 Constraints and Notes

Constraint: In a valid OrderLine instance, at most one of the attributes ServiceDescription and PartDescription shall be present.

6.10.2.5 OrderLinePartDescription Class

The OrderLinePartDescription class defines information about a part that is being requested in an order.

6.10.2.5.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PartType	PartTypeReference	0 to 1	A reference to the part type definition for the part being ordered.

Attribute/Role Name	Type	Multi- plicity	Description
Packaging	ItemPackaging	0 to 1	Information about how the part is packaged.
PartDetails	String	0 to 1	A description providing additional information about a part.

6.10.2.5.2 Superclasses & Inherited Attributes

None.

6.10.2.5.3 Constraints and Notes

Constraint: In a valid OrderLinePartDescription instance, at least one of the attributes PartType, Packaging, and PartDetails shall be present.

6.10.2.6 OrderLineServiceDescription Class

The OrderLineServiceDescription class defines information about a service that is being requested in an order.

6.10.2.6.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ServiceIdentifier	String	0 to 1	Identifying information that differentiates this service from other services.
ServiceDetails	String	0 to 1	A description providing additional information about a service.

6.10.2.6.2 Superclasses & Inherited Attributes

None.

6.10.2.6.3 Constraints and Notes

Constraint: In a valid OrderLineServiceDescription instance, at least one of the attributes ServiceIdentifier and ServiceDetails shall be present.

6.10.2.7 Schedule Class

The Schedule class defines a plan for the timing and duration of a collection of activities that are to occur during a specific time frame.

6.10.2.7.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
CreationTime	Timestamp	0 to 1	The time this schedule was created.

Attribute/Role Name	Type	Multi- plicity	Description
StartTime	Timestamp	0 to 1	The beginning of the time period covered by the schedule.
EndTime	Timestamp	0 to 1	The end of the time period covered by the schedule.
ScheduleItem	ScheduleItem	0 to *	A detailed description of an activity that is a part of a schedule.

6.10.2.7.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.10.2.7.3 Constraints and Notes

None.

6.10.2.8 ScheduleItem Class

The ScheduleItem class defines information about an activity that is a part of the collection of activities associated with a schedule.

6.10.2.8.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Status	JobStatus	1	An indicator of the progress made towards completing processing of the schedule item.
AssociatedJob	JobReference	0 to 1	Indicates that information relevant to the processing of this schedule item is defined in the referenced job.
AssociatedOrderInformation	OrderInformationReference	0 to 1	Indicates that information relevant to the processing of this schedule item is defined in the referenced order or order item.
AssociatedProcessPlan	ProcessPlanReference	0 to 1	Indicates that information relevant to the processing of this schedule item is defined in the referenced process plan.

Attribute/Role Name	Type	Multiplicity	Description
AssociatedProcess	ProcessReference	0 to 1	Indicates that information relevant to the processing of this schedule item is defined in the referenced process.
AssociatedMaintenancePlan	MaintenancePlanReference	0 to 1	Indicates that information relevant to the processing of this schedule item is defined in the referenced maintenance plan.
AssociatedMaintenanceProcess	MaintenanceProcessReference	0 to 1	Indicates that information relevant to the processing of this schedule item is defined in the referenced maintenance process.
PlannedEffort	ScheduleItemEffortDescription	0 to 1	The time frame during which processing of the schedule item is expected to take place, and the processing time, resource utilization, and material utilization expected to be required to complete processing of the schedule item.
ActualEffort	ScheduleItemEffortDescription	0 to 1	The time frame during which processing of the schedule item took place, and the processing time, resources, and materials used to complete processing of the schedule item.

6.10.2.8.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.10.2.8.3 Constraints and Notes

Constraint: At least one of the attributes PlannedEffort and ActualEffort shall be present.

Constraint: Valid instances of the ScheduleItem class shall not contain more than one of the following attributes:

AssociatedJob	AssociatedOrderInformation
AssociatedProcess	AssociatedMaintenancePlan
AssociatedProcessPlan	AssociatedMaintenanceProcess.

6.10.2.9 ScheduleItemEffortDescription Class

The ScheduleItemEffortDescription class defines information describing the time, resource, material, and process related aspects of a schedule item.

6.10.2.9.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
UpdateTime	Timestamp	0 to 1	The time of the last update of this schedule item's information.
StartTime	Timestamp	1	The time when the schedule item is to begin.
EndTime	Timestamp	0 to 1	The time when the schedule item should be done.
ProcessingTime	Duration	0 to 1	The amount of time between the starting and ending times that manufacturing activities related to this schedule item are being performed.
ResourcesUsed	ResourcesRequired	0 to *	The kind and quantity of resources used in the processing of this schedule item.
PartsUsed	PartGroup	0 to *	The parts that are used up or that become subcomponents of other parts as a result of processing this schedule item.
PartsProduced	PartGroup	0 to *	The parts that will be created by this schedule item.
Property	Property	0 to *	Name and value information describing a noteworthy characteristic of a schedule item.

6.10.2.9.2 Superclasses & Inherited Attributes

None.

6.10.2.9.3 Constraints and Notes

None.

6.11 Production Planning Package

The Production Planning package contains classes and relationships that can be used to create plans describing the dates and hours of operation for production resources and plans describing the sequence of processing steps necessary to manufacture products using the available production resources.

6.11.1 Diagrams

6.11.1.1 Calendar and Shift Related Classes

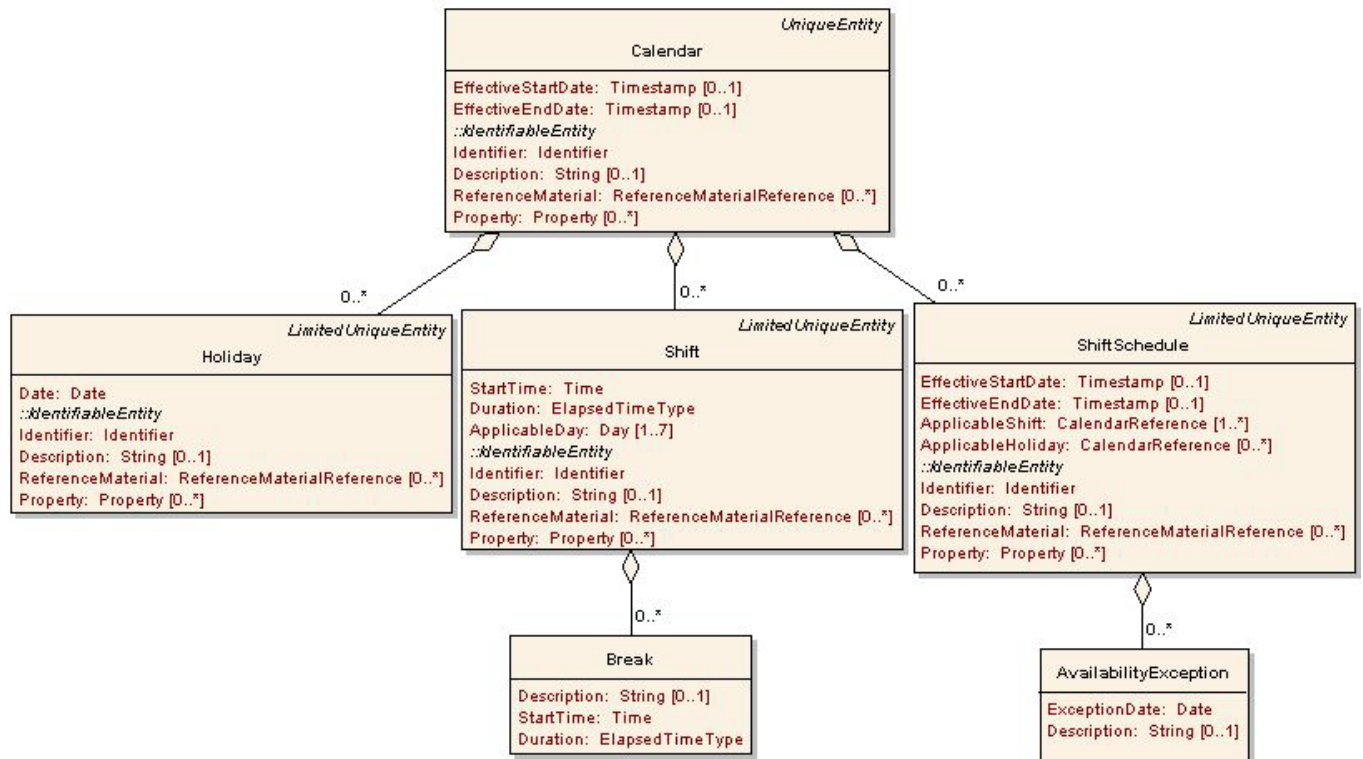


Figure 35: Calendar and Shift Related Classes

The Calendar class provides a means to define a manufacturing calendar for a production facility. A manufacturing calendar is a collection of information about the shifts, shift schedules, and holidays that apply to a facility over a specified time period. This class and its related class provide a flexible means to specify both times when resources are to be available for production activities and times when resources are not expected to be available.

The Shift class defines a period of time during which a resource should be available for production activities, and the days during a week that this time period is applicable.

The ShiftSchedule class defines information about a time period during which a number of shifts and holidays are applicable.

The Holiday class specifies days within the time period defined by a Calendar or ShiftSchedule when resources will not be available for production activities.

The Break class defines a time period within the time period defined by a Shift when resources are not available for production activities.

The AvailabilityException class defines a day within the time period defined by a ShiftSchedule when resources will not be available for production activities.

6.11.1.2 ProcessPlan, MaintenancePlan, and Related Classes

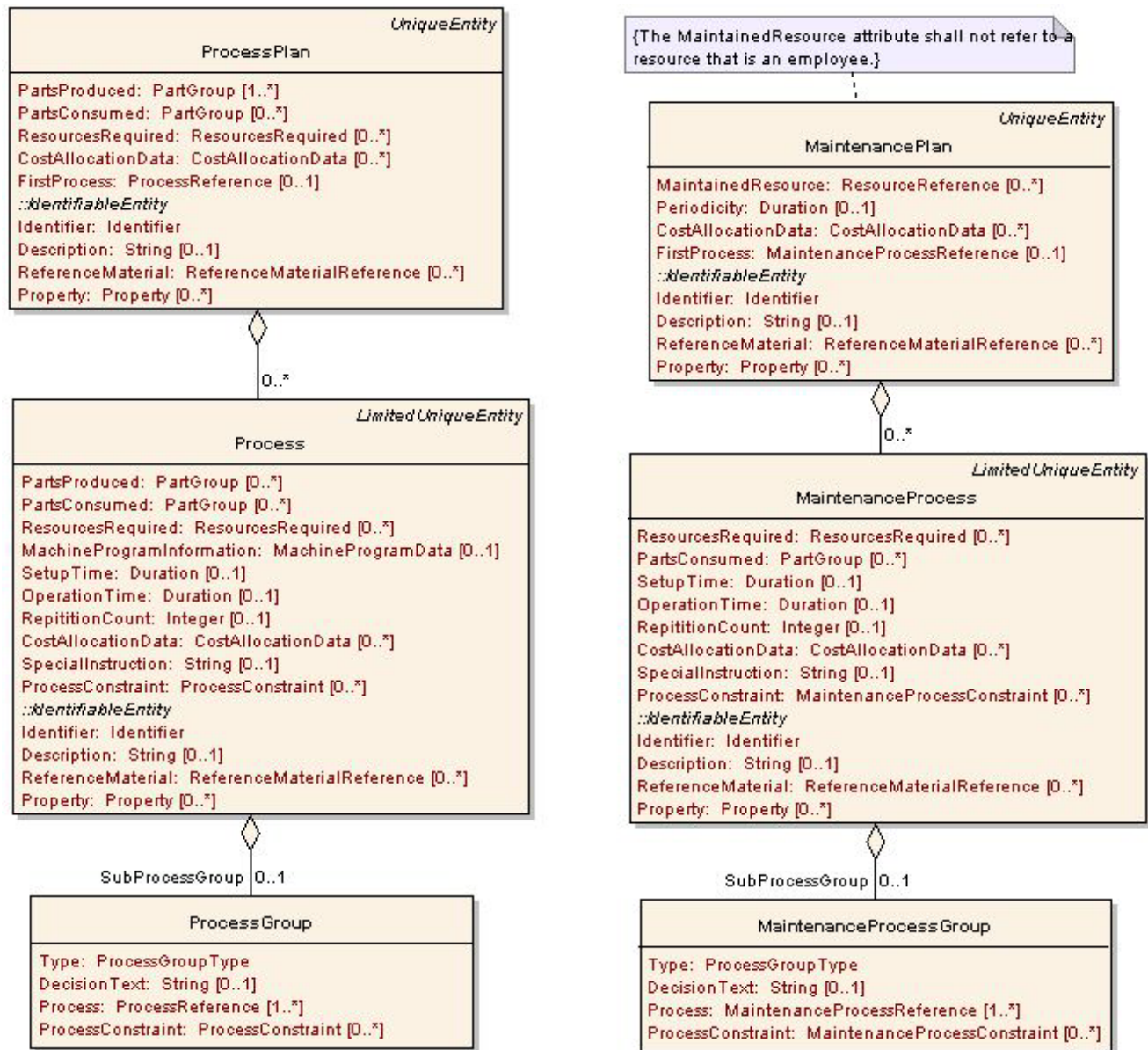


Figure 36: ProcessPlan, MaintenancePlan, and Related Classes

The ProcessPlan class provides a means to define a structured collection of information that describes a detailed strategy for creating a part. This collection includes information about the resources that will be used, the parts that will be produced, the sequence in which the resources will be used, and the sequence of manufacturing operation that will take place at each resource.

The Process class represents a manufacturing activity or group of related manufacturing activities that will be used to create a part. It includes information specifying which types of parts are produced or consumed,

which resources are needed, how long this activity will take, and if the processing specified in this activity is described in more detail in other activity definitions.

The ProcessGroup class defines a set of manufacturing activities that should be evaluated for execution as a group. It includes information specifying if the activities in the group should be executed sequentially, if the activities are intended to be executed concurrently, or if only one of the activities should be selected for execution based on a specified selection criterion.

The MaintenancePlan class provides a means to define a structured collection of information that describes a detailed strategy for maintaining a resource or group of resources. This collection includes information about the resources that will be maintained and how often the maintenance plan should be executed.

The MaintenanceProcess class represents a manufacturing activity or group of related manufacturing activities that will be used to maintain a resource or group of resources. It includes information specifying the resources to be maintained, the parts that may be used to maintain the resource, how long the activity or group of activities will take to complete, and whether the processing specified is described in more detail in other maintenance activity definitions.

The MaintenanceProcessGroup class defines a set of maintenance activities that should be evaluated for execution as a group. It includes information indicating if the activities in the group should be executed sequentially, if the activities are intended to be executed concurrently, or if only one of the activities should be selected for execution based on a specified selection criterion.

6.11.2 Classes

6.11.2.1 AvailabilityException Class

The AvailabilityException class defines information specifying days during a ShiftSchedule when resources should not be assigned production activities.

6.11.2.1.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi-plicity	Description
ExceptionDate	Date	1	The date for which resources should not be assigned production activities.
Description	String	0 to 1	The reason for the exception to the shift schedule.

6.11.2.1.2 Superclasses & Inherited Attributes

None.

6.11.2.1.3 Constraints and Notes

None.

6.11.2.2 Break Class

The Break class defines a time period during a Shift when resources should not be assigned production activities.

6.11.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Description	String	0 to 1	A description of the reason for the break.
StartTime	Time	1	The start of the time period.
Duration	ElapsedTimeType	1	The duration of the time period.

6.11.2.2.2 Superclasses & Inherited Attributes

None.

6.11.2.2.3 Constraints and Notes

None.

6.11.2.3 Calendar Class

The Calendar class defines information about the time periods in which resources may and may not be assigned production activities.

6.11.2.3.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
EffectiveStartDate	Timestamp	0 to 1	The start of the time period in which the calendar information is applicable.
EffectiveEndDate	Timestamp	0 to 1	The end of the time period in which the calendar information is applicable.
Shift	Shift	0 to *	The starting time and duration of a time period in which production activities may be assigned to resources and the days of the week on which this time period is applicable.
ShiftSchedule	ShiftSchedule	0 to *	A time period over which a group of shifts and holidays is applicable, and the specification of days during that time period (other than holidays) when resources will not be available for production activities.
Holiday	Holiday	0 to *	A day on which resources will not be available for production activities.

6.11.2.3.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.11.2.3.3 Constraints and Notes

None.

6.11.2.4 Holiday Class

The Holiday class defines information specifying days on which resources may not be assigned production activities.

6.11.2.4.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Date	Date	1	The date for which resources should not be assigned production activities.

6.11.2.4.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.11.2.4.3 Constraints and Notes

None.

6.11.2.5 MaintenancePlan Class

The MaintenancePlan class provides a means to define a detailed strategy for performing maintenance activities on a resource or resources. Information can be defined specifying the resources that will be maintained, the resources that will be used in the maintenance process, the parts that will be consumed by the process, and the sequence of maintenance operations that will take place.

6.11.2.5.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
MaintainedResource	ResourceReference	0 to *	A resource that this plan can be used to maintain.

Attribute/Role Name	Type	Multi- plicity	Description
Periodicity	Duration	0 to 1	An indication of how frequently this plan should be repeated to properly maintain its associated resources.
CostAllocationData	CostAllocationData	0 to *	Information describing, categorizing, and estimating the amount of costs that may be incurred as a result of performing this maintenance activity.
FirstProcess	MaintenanceProcessReference	0 to 1	The maintenance process associated with this maintenance plan that should be executed first.
MaintenanceProcesses	MaintenanceProcess	0 to *	A maintenance activity or group of related maintenance activities that will be used to maintain a resource.

6.11.2.5.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.11.2.5.3 Constraints and Notes

Constraint: The MaintainedResources attribute shall not refer to a resource of type “employee”.

6.11.2.6 MaintenanceProcess Class

The MaintenanceProcess class defines a manufacturing activity or group or manufacturing activities that are used to perform maintenance on resources. Information can be defined that describes the resources that will be used, the parts that will be consumed, the sequence in which resources will be used, and the sequence of maintenance activities within a group of maintenance activities.

6.11.2.6.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ResourcesRequired	ResourcesRequired	0 to *	The kind of and quantity of resources needed to execute this process.
PartsConsumed	PartGroup	0 to *	The parts that are used up as a result of executing this process.
SetupTime	Duration	0 to 1	The time required to prepare a resource to execute a process.

Attribute/Role Name	Type	Multi- plicity	Description
OperationTime	Duration	0 to 1	The time that is required for this maintenance activity to complete.
RepetitionCount	Integer	0 to 1	An indication that this maintenance activity should be repeated several times.
CostAllocationData	CostAllocationData	0 to *	Information describing, categorizing, and estimating the amount of costs that may be incurred as a result of executing this maintenance activity.
SpecialInstruction	String	0 to 1	Information highlighting some aspect of this process that is atypical, unusual, or unique, and that may affect how the maintenance activities defined by this process should be carried out.
ProcessConstraint	MaintenanceProcessConstraint	0 to *	A restriction on when a maintenance activity may start or complete based on when a related maintenance activity starts or completes.
SubProcessGroup	MaintenanceProcessGroup	0 to 1	A group of related maintenance activities that define/refine at a greater level of detail the processing specified for this maintenance activity.

6.11.2.6.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.11.2.6.3 Constraints and Notes

None.

6.11.2.7 MaintenanceProcessGroup Class

The MaintenanceProcessGroup class defines a set of maintenance activities that should be evaluated for execution as a group.

6.11.2.7.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
Type	ProcessGroupType	1	Specifies whether the maintenance processes in the group should be executed sequentially, whether they may be executed concurrently, or whether only one maintenance process in the group should be selected for execution.
DecisionText	String	0 to 1	Specifies how the maintenance process to be executed should be chosen for decision type maintenance process groups.
Process	MaintenanceProcessReference	1 to *	A maintenance process that is a part of this maintenance process group.
ProcessConstraint	MaintenanceProcessConstraint	0 to *	A restriction on when a process may start or complete based on when a related process starts or completes.

6.11.2.7.2 Superclasses & Inherited Attributes

None.

6.11.2.7.3 Constraints and Notes

None.

6.11.2.8 Process Class

The Process class defines a manufacturing activity or group or manufacturing activities that are a part of a detailed strategy for creating a part. Information can be defined that describes the resources that will be used, the parts that will be consumed and produced, the sequence in which resources will be used, and the sequence of activities within a group of activities.

6.11.2.8.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
PartsProduced	PartGroup	0 to *	The parts that will be created as a result of executing this process.
PartsConsumed	PartGroup	0 to *	The parts that are used up or that become subcomponents of other parts as a result of executing this process.
ResourcesRequired	ResourcesRequired	0 to *	The kind of and quantity of resources needed to execute this process.

Attribute/Role Name	Type	Multi- plicity	Description
MachineProgramInformation	MachineProgramData	0 to 1	Information about a machine control program that will be run as a part of this process.
SetupTime	Duration	0 to 1	The time required to prepare a resource to execute a process.
OperationTime	Duration	0 to 1	The time that is required for this process to complete.
RepetitionCount	Integer	0 to 1	An indication that this process should be repeated several times.
CostAllocationData	CostAllocationData	0 to *	Information describing, categorizing, and estimating the amount of costs that may be incurred as a result of executing this process.
SpecialInstruction	String	0 to 1	Information highlighting some aspect of this process that is atypical, unusual, or unique, and that may affect how the production activities defined by this process should be carried out.
ProcessConstraint	ProcessConstraint	0 to *	A restriction on when a process may start or complete based on when a related process starts or completes.
SubProcessGroup	ProcessGroup	0 to 1	A group of related manufacturing activities that define/refine at a greater level of detail the processing specified for this process.

6.11.2.8.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.11.2.8.3 Constraints and Notes

None.

6.11.2.9 ProcessGroup Class

The ProcessGroup class defines a set of manufacturing activities that should be evaluated for execution as a group.

6.11.2.9.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
Type	ProcessGroupType	1	Specifies whether the processes in the group should be executed sequentially, whether they may be executed concurrently, or whether only one process in the group should be selected for execution.
DecisionText	String	0 to 1	Specifies how the process to be executed should be chosen for decision type process groups.
Process	ProcessReference	1 to *	A process that is a part of this process group.
ProcessConstraint	ProcessConstraint	0 to *	A restriction on when a process may start or complete based on when a related process starts or completes.

6.11.2.9.2 Superclasses & Inherited Attributes

None.

6.11.2.9.3 Constraints and Notes

None.

6.11.2.10 ProcessPlan Class

The ProcessPlan class provides a means to define a detailed strategy for creating a part. Information about the resources that will be used, the parts that will be consumed and produced, the sequence in which resources will be used, and the sequence of manufacturing operations that will take place at each resource can be defined using this class.

6.11.2.10.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
PartsProduced	PartGroup	1 to *	The parts that will be created as a result of executing this process plan.
PartsConsumed	PartGroup	0 to *	The parts that are used up or that become subcomponents of other parts as a result of executing this process plan.
ResourcesRequired	ResourcesRequired	0 to *	The kind of and quantity of resources needed to execute this process plan.

Attribute/Role Name	Type	Multi- plicity	Description
CostAllocationData	CostAllocationData	0 to *	Information describing, categorizing, and estimating the amount of costs that may be incurred as a result of executing this process plan.
FirstProcess	ProcessReference	0 to 1	The process associated with this process plan that should be executed first.
Process	Process	0 to *	A manufacturing activity or group of related manufacturing activities that will be used to create a part.

6.11.2.10.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.11.2.10.3 Constraints and Notes

None.

6.11.2.11 Shift Class

The Shift class defines a time period when resources may be assigned production activities and the days on which that time period applies.

6.11.2.11.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
StartTime	Time	1	The start of the time period when resources may be assigned production activities.
Duration	ElapsedTimeType	1	The duration of the time period.
ApplicableDay	Day	1 to 7	A day of the week that the start of the time period applies to.
Break	Break	0 to *	A short time during the time period defined for the Shift during which resources should not be assigned production activities.

6.11.2.11.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.11.2.11.3 Constraints and Notes

None.

6.11.2.12 ShiftSchedule Class

The ShiftSchedule class defines information about a time period during which a number of Shifts and Holidays are applicable. Information about exceptions to the ShiftSchedule, days during the ShiftSchedule's time period when resources should not be assigned production activities, can also be defined.

6.11.2.12.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
EffectiveStartDate	Timestamp	0 to 1	The start of the time period for which the ShiftSchedule is applicable.
EffectiveEndDate	Timestamp	0 to 1	The end of the time period for which the ShiftSchedule is applicable.
ApplicableShift	CalendarReference	1 to *	A Shift that is a part of the ShiftSchedule.
ApplicableHoliday	CalendarReference	0 to *	A Holiday that is a part of the ShiftSchedule.
AvailabilityException	AvailabilityException	0 to *	A day during the time period defined for the ShiftSchedule when resources should not be assigned production activities.

6.11.2.12.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.11.2.12.3 Constraints and Notes

None.

6.12 Resource Information Package

The Resource Information package contains classes for creating definitions of the characteristics and capabilities of the equipment and employees used in the manufacturing process, the skills associated with employees, and setup information required for the making efficient use of non-employee resources.

6.12.1 Diagrams

6.12.1.1 The Resource Class and ResourceClass Class

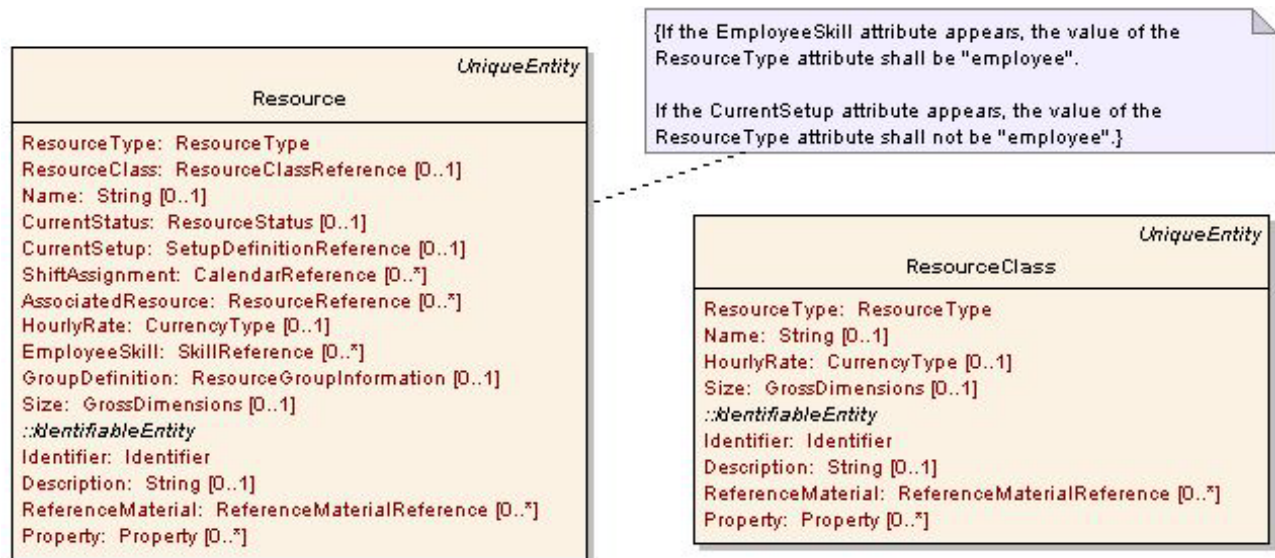


Figure 37: The Resource Class and ResourceClass Class

The Resource class defines information about a piece of equipment or an employee, or a collection of pieces of equipment and/or employees, that are used in the manufacturing process. All resources have a type describing in general the kind of manufacturing asset or assets the resource represents. The resource types include machine, station, employee, conveyor, fixture, tool, and other. Other information that can be defined for a resource includes the current operational state of the resource, the gross dimensions of the resource, the “class” to which a resource belongs, and information about the group members for resources that are resource groups.

The ResourceClass class provides a means to create a classification scheme for resources based on descriptions of the characteristics that those resources possess. A name and identifier for each class of resource can be defined, and characteristics for resources in the class can be defined by using property definitions.

6.12.1.2 ResourceGroupInformation and Related Classes

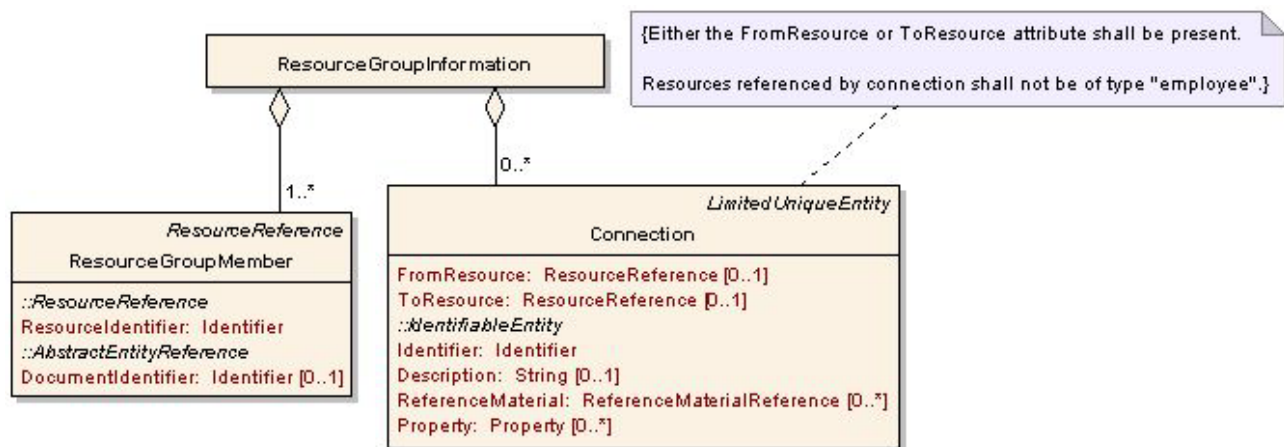


Figure 38: ResourceGroupInformation and Related Classes

The ResourceGroupInformation class defines information about a group of resources that may be considered a singular resource to facilitate efficient resource assignment for production operations. Associated with a ResourceGroupInformation instance may be several instances of the ResourceGroupMember class, indicating which resources are members of the resource group, and several instances of the Connection class, indicating logical flow relationships between the resources in the group and other resources.

6.12.1.3 The SkillDefinition Class and SkillLevel Class

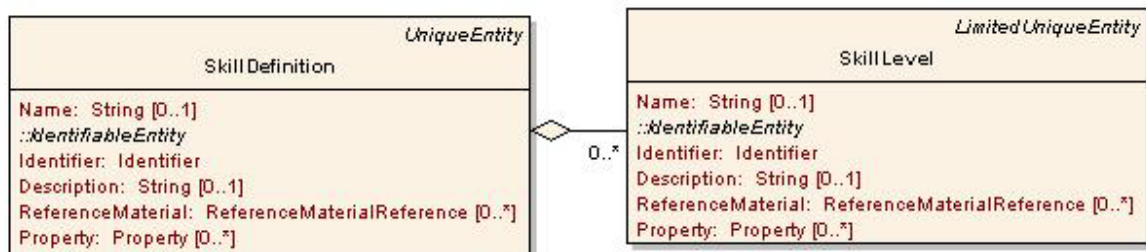


Figure 39: The SkillDefinition Class and SkillLevel Class

The SkillDefinition and SkillLevel classes provide a means to describe the abilities and/or levels of expertise that an employee resource must possess to be able to perform a manufacturing task. This information can then be used to enhance the definitions of resources and resource classes for employee resources.

6.12.1.4 SetupDefinition and Related Classes

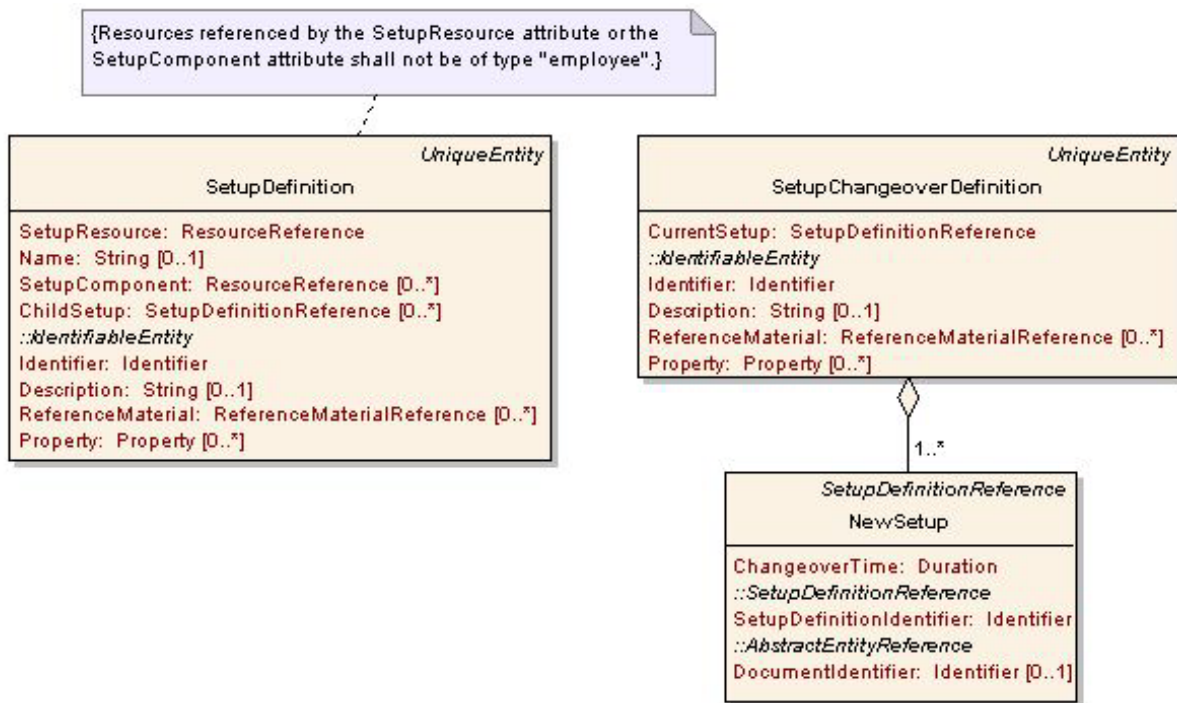


Figure 40: SetupDefinition and Related Classes

The SetupDefinition class provides a means to describe the particular configuration a resource should be in so that it is able to perform a manufacturing activity, and the additional resources that need to be present to perform that activity. This information can then be used to enhance the description of a non-employee resource's current state and to facilitate the efficient assignment of appropriate production activities to resources.

The SetupChangeoverDefinition class provides a means to define detailed information about the time it takes to change a resource from one setup state to another setup state. This class identifies a specific SetupDefinition, and indicates the time to change from that setup to other setups through association with multiple instances of the NewSetup class. Each NewSetup class instance specifies the time to change a resource from its current setup to a specified new setup.

6.12.2 Classes

6.12.2.1 Connection Class

The Connection class defines information about how parts flow from one resource to another resource.

6.12.2.1.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
FromResource	ResourceReference	0 to 1	Indicates that parts may flow from this resource to another resource.

Attribute/Role Name	Type	Multi- plicity	Description
ToResource	ResourceReference	0 to 1	Indicates that parts may flow to this resource from another resource.

6.12.2.1.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.12.2.1.3 Constraints and Notes

Constraint: At least one of the attributes FromResource and ToResource shall be present in valid instances of the Connection class.

Constraint: The resources referenced by FromResource and ToResource shall not be of type employee.

6.12.2.2 NewSetup Class

The NewSetup class defines information specifying how long it takes to change to a specified setup.

6.12.2.2.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ChangeoverTime	Duration	1	The time it takes to change a resource to the referenced setup from the associated current setup.

6.12.2.2.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>SetupDefinitionReference</i> (subclass of <i>SimpleReference</i>)	SetupDefinitionIdentifier, DocumentIdentifier.

6.12.2.2.3 Constraints and Notes

None.

6.12.2.3 Resource Class

The Resource class defines information about the equipment and workers that are used to carry out production activities.

6.12.2.3.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ResourceType	ResourceType	1	The general kind of resource that this resource is.
ResourceClass	ResourceClassReference	0 to 1	A reference to a distinct category for resources with similar characteristics.
Name	String	0 to 1	The name of the resource.
CurrentStatus	ResourceStatus	0 to 1	An indication as to whether the resource is currently performing or is currently able to perform production activities.
CurrentSetup	SetupDefinitionReference	0 to 1	Information about how the resource has been configured to carry out production activities.
ShiftAssignment	CalendarReference	0 to *	Information defining the days of the week and the time period during those days when the resource is expected to be available for production activities.
AssociatedResource	ResourceReference	0 to *	Information specifying another resource that is expected to perform production activities in concert with this resource.
HourlyRate	CurrencyType	0 to 1	The per hour expense incurred by operating this resource.
EmployeeSkill	SkillReference	0 to *	Training, competence, or an ability possessed by an employee resource that makes him or her suitable for some specific production activity.
GroupDefinition	ResourceGroupInformation	0 to 1	An indication that this resource defines information about a group of resource instances that work together to perform production activities, and information about the relationships between those resources.
Size	GrossDimensions	0 to 1	The approximate size of the resource.

6.12.2.3.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.12.2.3.3 Constraints and Notes

Constraint: If the EmployeeSkill attribute is used, the value of ResourceType shall be “employee”.

Constraint: If the CurrentSetup attribute is used, the value of ResourceType shall not be “employee”.

6.12.2.4 ResourceClass Class

The ResourceClass class defines information about a category to which resources may belong and the characteristics that all resources in that category possess.

6.12.2.4.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ResourceType	ResourceType	1	The general kind of resource that resources of this resource class are.
Name	String	0 to 1	The name of the resource class.
HourlyRate	CurrencyType	0 to 1	The per hour expense incurred by operating a resource of this class.
Size	GrossDimensions	0 to 1	The approximate size of the resource.

6.12.2.4.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.12.2.4.3 Constraints and Notes

None.

6.12.2.5 ResourceGroupInformation Class

The ResourceGroupInformation class defines information about the members of a resource group and part flow relationships between resources.

6.12.2.5.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
ResourceGroupMember	ResourceGroupMember	1 to *	A reference to a resource that is a member of a resource group.
Connection	Connection	0 to *	Information describing how parts flow from one resource to another.

6.12.2.5.2 Superclasses & Inherited Attributes

None.

6.12.2.5.3 Constraints and Notes

None.

6.12.2.6 ResourceGroupMember Class

The ResourceGroupMember class defines information that indicates which resources are members of a resource group.

6.12.2.6.1 Defined Attributes and Association Roles

None.

6.12.2.6.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>ResourceReference</i> (subclass of <i>SimpleReference</i>)	ResourceIdentifier, DocumentIdentifier.

6.12.2.6.3 Constraints and Notes

None.

6.12.2.7 SetupChangeoverDefinition Class

The SetupChangeoverDefinition class defines information specifying how long it takes to change a resource's configuration from one setup to another setup.

6.12.2.7.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
CurrentSetup	SetupDefinitionReference	1	The setup that can be changed from.
NewSetup	NewSetup	1 to *	A setup that can be changed to.

6.12.2.7.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.12.2.7.3 Constraints and Notes

None.

6.12.2.8 SetupDefinition Class

The SetupDefinition class defines information about a specific way that a resource can be configured to enable efficient assignment and scheduling of production activities.

6.12.2.8.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multi- plicity	Description
SetupResource	ResourceReference	1	The resource for which this setup is being created.
Name	String	0 to 1	The name of the setup.
SetupComponent	ResourceReference	0 to *	A different resource that is a part of the setup for this resource.
ChildSetup	SetupDefinitionReference	0 to *	A separate setup definition that describes part of the configuration for this setup definition.

6.12.2.8.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.12.2.8.3 Constraints and Notes

Constraint: The SetupResource and SetupComponent attributes shall refer to a non-employee resource.

6.12.2.9 SkillDefinition Class

The SkillDefinition class defines information about an ability or special training that an employee may possess that will allow him or her to perform production activities.

6.12.2.9.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
Name	String	0 to 1	The name of the skill.

6.12.2.9.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>UniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.12.2.9.3 Constraints and Notes

None.

6.12.2.10 SkillLevel Class

The SkillLevel class defines information about a level of expertise in a skill.

6.12.2.10.1 Defined Attributes and Association Roles

Attribute/Role Name	Type	Multiplicity	Description
Name	String	0 to 1	The name of the skill level.

6.12.2.10.2 Superclasses & Inherited Attributes

Superclass Name	Inherited attributes
<i>LimitedUniqueEntity</i> (subclass of <i>IdentifiableEntity</i>)	Identifier, Description, ReferenceMaterial, Property.

6.12.2.10.3 Constraints and Notes

None.

6.13 Support Package

The Support package defines packages that contain definitions for simple types and other basic structures that are used in other packages to define more complex structures. Although it does not directly define any classes, several nested subpackages are defined within the scope of the Support package. It is within these nested packages that the simple types and structures that are reused for class and relationship definition in other CMSD packages are directly defined. Each of the Support package's subpackages defines a focused, cohesive set of classes and relationships that, through reuse, facilitate the definition of consistent structures within all of the packages of CMSD.

6.13.1 Diagrams

6.13.1.1 Support Package Sub-packages

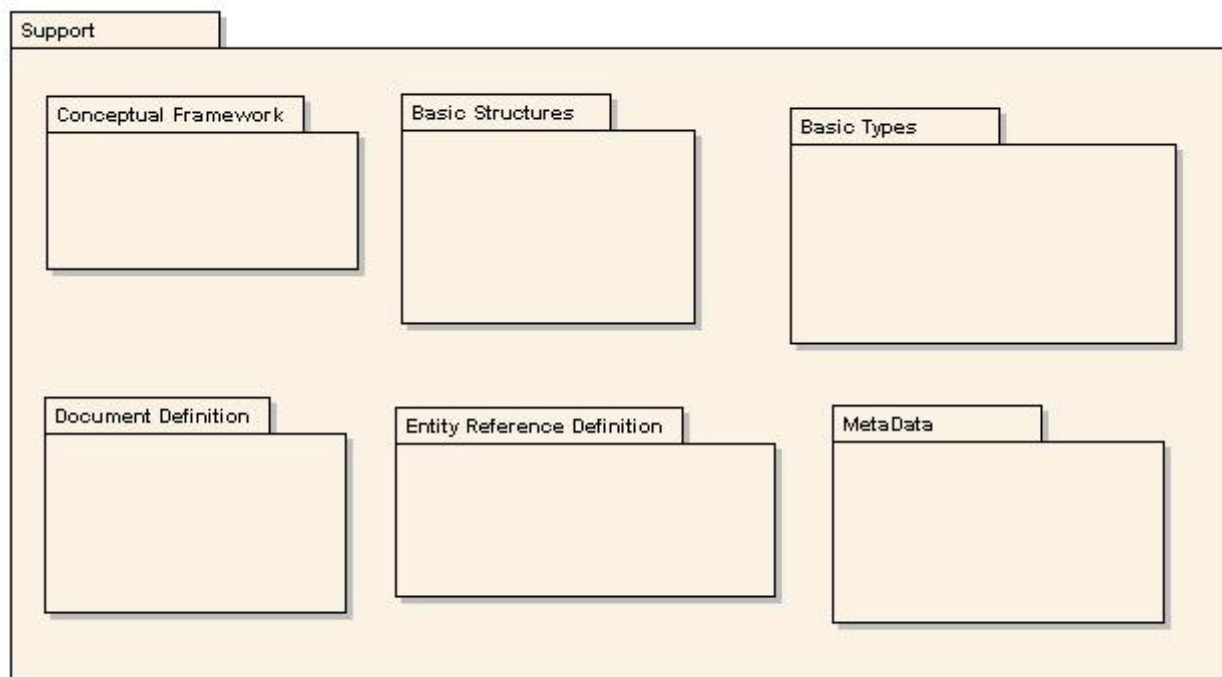


Figure 41 - The Support Package and Its Sub-packages

The Conceptual Framework package contains classes and relationships that provide an abstract framework that describes how important groupings of CMSD information, referred to as entities, relate to each other, and under what circumstances those entities may be considered unique.

The Basic Structures package contains classes and relationships defining simple structures intended to support the definition of more complex structures in other CMSD packages.

The Basic Types package contains definitions for enumerations and data types that can be used in the definition of class attributes for classes defined in other CMSD packages.

The Document Definition package describes the structure and allowable content for a CMSD “document”. In CMSD, a document is an assemblage of data components, defined according to CMSD class and relationship definitions, that has been assembled with some specific purpose in mind, and that is suitable for archival or exchange.

The Entity Reference Definition package contains class definitions that facilitate the ability of instances of CMSD classes to unambiguously refer to other CMSD class instances.

The Metadata package contains classes and relationships that provide a means to describe the intended meaning and allowable content for the “property” attributes that are a part of many CMSD classes.

6.13.2 Classes

No classes are directly defined by the Support package. Each of the subpackages of the Support package is described separately in other sections of this document.

Annex A Bibliography

(Informative)

[A1] Lee, Y. T., and McLean, C. R., "A Neutral Data Interface Specification for Simulating Machine Shop Operations," *Production Planning & Control*, vol. 17, no.2, pp. 143-154, March 2006.

[A2] Lu, R. F., Qiao, G., Riddick, F. H., and McLean, C. R., "NIST XML Simulation Interface Specification at Boeing: A Case Study," 2003 Winter Simulation Conference Proceedings.

[A3] McLean, C., Riddick, F., and Lee, Y. T., "An Architecture and Interfaces for Distributed Manufacturing Simulation," *SIMULATION*, vol. 81, iss.1, pp.15-32, January 2005.

[A4] NISTIR 7198, Shop Data Model and Interface Specification, January 2005.

[A5] Riddick, F. and Lee, Y. T., "Core Manufacturing Simulation Data Information Model (CMSDIM), Part1: UML Model," SISO SAC-PDG-CMSD Discussions (on-line), September 2006.

[A6] SISO-PDG-PN-CMSD, "Core Manufacturing Simulation Data (CMSD) Product Nomination (PN)," 2004.

Annex B Class Cross Reference**(Informative)**

The table below alphabetically lists each class defined in CMSD, and where in the document information about that class is located. The fields of the table are defined as follows:

- **Class Name:** The name of the CMSD class
- **Section Number:** The section in the document where the description of the class can be found.
- **Page Number:** The page number where the description of the class can be found.
- **Diagram:** The figure number of the diagram where a visual representation of the class can be found.
- **Package Name:** The name of the UML package that contains the class.

Class Name	Section Number	Page Number	Diagram	Package Name
AbstractEntityReference	6.6.2.1	81	Figure 20	Entity Reference Definition
Address	6.1.2.1	25	Figure 6	Basic Structures
AreaType	6.1.2.2	26	Figure 3	Basic Structures
AreaUnit	6.2.2.1	56	Figure 13	Basic Types
AvailabilityException	6.11.2.1	133	Figure 35	Production Planning
BaseLocation	6.2.2.2	56	Figure 15	Basic Types
Basic Entity Reference Classes	6.6.2.2	81	Figure 20	Entity Reference Definition
BasicShape	6.7.2.1	93	Figure 27	Layout
BasicShapeType	6.2.2.3	57	Figure 15	Basic Types
BillOfMaterials	6.9.2.1	113	Figure 31	Part Information
BillOfMaterialsComponent	6.9.2.2	114	Figure 31	Part Information
BillOfMaterialsComponentReference	6.6.2.3	82	Figure 21	Entity Reference Definition
BillOfMaterialsReference	6.6.2.2	81	Figure 20	Entity Reference Definition
Boolean	6.2.2.4	57	Figure 12	Basic Types

Class Name	Section Number	Page Number	Diagram	Package Name
BoundaryDefinition	6.1.2.3	26	Figure 9	Basic Structures
Box	6.7.2.2	93	Figure 27	Layout
Break	6.11.2.2	133	Figure 35	Production Planning
Calendar	6.11.2.3	134	Figure 35	Production Planning
CalendarReference	6.6.2.4	83	Figure 21	Entity Definition Reference Definition
CardinalitySpecification	6.2.2.5	57	Figure 12	Basic Types
Circle	6.7.2.3	94	Figure 27	Layout
CMSDDocument	6.5.2.1	72	Figure 19	Document Definition
CMSDDocumentReference	6.5.2.2	72	Figure 19	Document Definition
ColorDefinition	6.2.2.6	57	Figure 15	Basic Types
ColorHexString	6.2.2.7	57	Figure 15	Basic Types
ColorHighlight	6.1.2.4	27	Figure 9	Basic Structures
ColorName	6.2.2.8	57	Figure 15	Basic Types
CommunicationMethod	6.1.2.5	28	Figure 6	Basic Structures
Connection	6.12.2.1	145	Figure 38	Resource Information
ConnectionType	6.2.2.9	57	Figure 16	Basic Types
ContactInformation	6.1.2.6	28	Figure 6	Basic Structures
ContactParty	6.1.2.7	28	Figure 6	Basic Structures
Coordinate2D	6.1.2.8	29	Figure 9	Basic Structures
Coordinate3D	6.1.2.9	29	Figure 9	Basic Structures
CoordinateSystem	6.2.2.10	58	Figure 15	Basic Types
CostAllocationData	6.1.2.10	29	Figure 8	Basic Structures
CostCategory	6.2.2.11	58	Figure 16	Basic Types
CostType	6.2.2.12	58	Figure 16	Basic Types

Class Name	Section Number	Page Number	Diagram	Package Name
CurrencyType	6.1.2.11	30	Figure 3	Basic Structures
CurrencyUnit	6.2.2.13	58	Figure 13	Basic Types
CurvedSegment	6.7.2.4	94	Figure 28	Layout
DataSection	6.5.2.3	73	Figure 19	Document Definition
Date	6.2.2.14	58	Figure 12	Basic Types
Day	6.2.2.15	59	Figure 16	Basic Types
Decimal	6.2.2.16	59	Figure 12	Basic Types
Distribution	6.1.2.12	31	Figure 5	Basic Structures
DistributionDefinition	6.1.2.13	31	Figure 5	Basic Structures
DistributionDefinitionReference	6.6.2.2	81	Figure 20	Entity Reference Definition
DistributionParameter	6.1.2.14	32	Figure 5	Basic Structures
Duration	6.1.2.15	32	Figure 5	Basic Structures
ElapsedTimeType	6.1.2.16	33	Figure 3	Basic Structures
Email	6.1.2.17	34	Figure 6	Basic Structures
Entity	6.4.2.1	68	Figure 18	Conceptual Framework
Event	6.1.2.18	34	Figure 8	Basic Structures
GraphicDescription	6.7.2.5	95	Figure 25	Layout
GraphicDescriptionType	6.2.2.17	59	Figure 15	Basic Types
GrossDimensions	6.1.2.19	35	Figure 3	Basic Structures
HeaderSection	6.5.2.4	74	Figure 19	Document Definition
Holiday	6.11.2.4	135	Figure 35	Production Planning
IdentifiableEntity	6.4.2.2	68	Figure 18	Conceptual Framework
Identifier	6.2.2.18	59	Figure 12	Basic Types
ImageGraphic	6.7.2.6	96	Figure 25	Layout

Class Name	Section Number	Page Number	Diagram	Package Name
ImageResolution	6.1.2.20	35	Figure 9	Basic Structures
Integer	6.2.2.19	59	Figure 12	Basic Types
InventoryItem	6.9.2.3	115	Figure 32	Part Information
InventoryItemClass	6.9.2.4	116	Figure 32	Part Information
InventoryItemClassReference	6.6.2.2	81	Figure 20	Entity Definition Reference Definition
InventoryItemReference	6.6.2.2	81	Figure 20	Entity Definition Reference Definition
InventoryItemType	6.2.2.20	59	Figure 16	Basic Types
ItemPackaging	6.1.2.21	36	Figure 8	Basic Structures
Job	6.10.2.1	122	Figure 33	Production Operations
JobConstraint	6.1.2.22	36	Figure 7	Basic Structures
JobEffortDescription	6.10.2.2	123	Figure 33	Production Operations
JobReference	6.6.2.2	81	Figure 20	Entity Definition Reference Definition
JobStatus	6.2.2.21	59	Figure 16	Basic Types
Layout	6.7.2.7	96	Figure 22	Layout
LayoutElement	6.7.2.8	97	Figure 22	Layout
LayoutElementReference	6.6.2.2	81	Figure 20	Entity Definition Reference Definition
LayoutLengthUnit	6.2.2.23	60	Figure 15	Basic Types
LayoutObject	6.7.2.9	98	Figure 22	Layout
LengthType	6.1.2.23	37	Figure 3	Basic Structures
LengthUnit	6.2.2.22	59	Figure 13	Basic Types
LimitedUniqueEntity	6.4.2.3	69	Figure 18	Conceptual Framework
LocationDefinition	6.1.2.24	37	Figure 11	Basic Structures

Class Name	Section Number	Page Number	Diagram	Package Name
LotInformation	6.1.2.25	38	Figure 11	Basic Structures
MachineProgramData	6.1.2.26	38	Figure 8	Basic Structures
MaintenancePlan	6.11.2.5	135	Figure 36	Production Planning
MaintenancePlanReference	6.6.2.2	81	Figure 20	Entity Reference Definition
MaintenanceProcess	6.11.2.6	136	Figure 36	Production Planning
MaintenanceProcessConstraint	6.1.2.27	39	Figure 7	Basic Structures
MaintenanceProcessGroup	6.11.2.7	137	Figure 36	Production Planning
MaintenanceProcessReference	6.6.2.5	83	Figure 21	Entity Reference Definition
Metadata	6.5.2.5	75	Figure 19	Document Definition
ModelGraphic	6.7.2.10	98	Figure 25	Layout
NewSetup	6.12.2.2	146	Figure 40	Resource Information
NonNegativeInteger	6.2.2.24	60	Figure 12	Basic Types
Order	6.10.2.3	124	Figure 33	Production Operations
OrderInformationReference	6.6.2.6	84	Figure 21	Entity Reference Definition
OrderLine	6.10.2.4	125	Figure 33	Production Operations
OrderLinePartDescription	6.10.2.5	126	Figure 33	Production Operations
OrderLineServiceDescription	6.10.2.6	127	Figure 33	Production Operations
OrderStatus	6.2.2.25	60	Figure 16	Basic Types
Part	6.9.2.5	117	Figure 30	Part Information
PartGroup	6.1.2.28	39	Figure 11	Basic Structures
PartProductionStatus	6.2.2.26	60	Figure 16	Basic Types
PartReference	6.6.2.2	81	Figure 20	Entity Reference Definition

Class Name	Section Number	Page Number	Diagram	Package Name
PartType	6.9.2.6	118	Figure 30	Part Information
PartTypeReference	6.6.2.2	81	Figure 20	Entity Reference Definition
PerMeasuredAmountPackaging	6.1.2.29	40	Figure 8	Basic Structures
PerPiecePackaging	6.1.2.30	41	Figure 8	Basic Structures
Phone	6.1.2.31	41	Figure 6	Basic Structures
Placement	6.7.2.11	99	Figure 22	Layout
Polygon	6.7.2.12	99	Figure 27	Layout
PowerType	6.1.2.32	41	Figure 3	Basic Structures
PowerUnit	6.2.2.27	60	Figure 13	Basic Types
PrecedenceConstraint	6.1.2.33	42	Figure 7	Basic Structures
PrecedenceRelationship	6.2.2.28	60	Figure 16	Basic Types
Process	6.11.2.8	138	Figure 36	Production Planning
ProcessConstraint	6.1.2.34	43	Figure 7	Basic Structures
ProcessGroup	6.11.2.9	139	Figure 36	Production Planning
ProcessGroupType	6.2.2.29	61	Figure 16	Basic Types
ProcessPlan	6.11.2.10	140	Figure 36	Production Planning
ProcessPlanReference	6.6.2.2	81	Figure 20	Entity Reference Definition
ProcessReference	6.6.2.7	85	Figure 21	Entity Reference Definition
Property	6.1.2.35	43	Figure 4	Basic Structures
PropertyCardinality	6.8.2.1	108	Figure 29	Metadata
PropertyDataTypeDescription	6.8.2.2	108	Figure 29	Metadata
PropertyDescription	6.8.2.3	109	Figure 29	Metadata



Class Name	Section Number	Page Number	Diagram	Package Name
PropertyDescriptionReference	6.6.2.2	81	Figure 20	Entity Reference Definition
PropertyExtensibleEntity	6.2.2.30	61	Figure 14	Basic Types
PropertyType	6.2.2.31	61	Figure 14	Basic Types
QuantityType	6.1.2.36	44	Figure 3	Basic Structures
ReferenceMaterial	6.1.2.37	44	Figure 10	Basic Structures
ReferenceMaterialReference	6.6.2.2	81	Figure 20	Entity Reference Definition
ReferenceProperty	6.1.2.38	45	Figure 4	Basic Structures
ReferenceTypeName	6.2.2.32	61	Figure 14	Basic Types
ReferencingEntity	6.4.2.4	70	Figure 18	Conceptual Framework
RequiredApplication	6.1.2.39	46	Figure 10	Basic Structures
Resource	6.12.2.3	146	Figure 37	Resource Information
ResourceClass	6.12.2.4	148	Figure 37	Resource Information
ResourceClassReference	6.6.2.2	81	Figure 20	Entity Reference Definition
ResourceGroupInformation	6.12.2.5	148	Figure 38	Resource Information
ResourceGroupMember	6.12.2.6	149	Figure 38	Resource Information
ResourceReference	6.6.2.2	81	Figure 20	Entity Reference Definition
ResourcesRequired	6.1.2.40	46	Figure 11	Basic Structures
ResourceStatus	6.2.2.33	61	Figure 16	Basic Types
ResourceType	6.2.2.34	62	Figure 16	Basic Types
Rotation	6.7.2.13	100	Figure 24	Layout
Scaling	6.7.2.14	100	Figure 24	Layout
Schedule	6.10.2.7	127	Figure 34	Production Operations

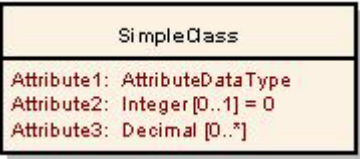
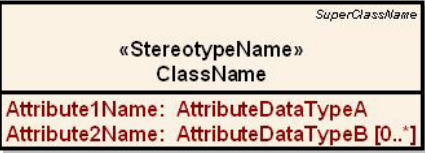
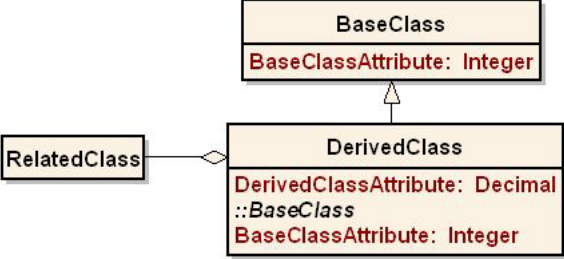
Class Name	Section Number	Page Number	Diagram	Package Name
ScheduleInformationReference	6.6.2.8	85	Figure 21	Entity Reference Definition
ScheduleItem	6.10.2.8	128	Figure 34	Production Operations
ScheduleItemEffortDescription	6.10.2.9	129	Figure 34	Production Operations
SegmentShape	6.7.2.15	101	Figure 28	Layout
SegmentType	6.2.2.35	62	Figure 15	Basic Types
SetupChangeoverDefinition	6.12.2.7	149	Figure 40	Resource Information
SetupChangeoverReference	6.6.2.2	81	Figure 20	Entity Reference Definition
SetupDefinition	6.12.2.8	150	Figure 40	Resource Information
SetupDefinitionReference	6.6.2.2	81	Figure 20	Entity Reference Definition
ShapeDescription	6.7.2.16	102	Figure 23	Layout
ShapeDescriptionType	6.2.2.36	62	Figure 15	Basic Types
ShapeLabelDefinition	6.1.2.41	47	Figure 9	Basic Structures
Shift	6.11.2.11	141	Figure 35	Production Planning
ShiftSchedule	6.11.2.12	142	Figure 35	Production Planning
SimpleDataTypeName	6.2.2.37	62	Figure 14	Basic Types
SimpleProperty	6.1.2.42	48	Figure 4	Basic Structures
SkillDefinition	6.12.2.9	151	Figure 39	Resource Information
SkillLevel	6.12.2.10	151	Figure 39	Resource Information
SkillReference	6.6.2.9	86	Figure 21	Entity Reference Definition
SpatialDimension	6.1.2.43	48	Figure 9	Basic Structures
SpeedType	6.1.2.44	49	Figure 3	Basic Structures
SpeedUnit	6.2.2.38	62	Figure 13	Basic Types

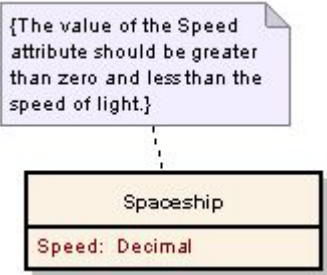
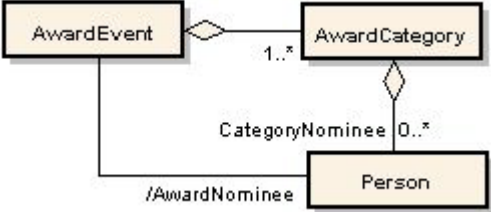

Class Name	Section Number	Page Number	Diagram	Package Name
StochasticProperty	6.1.2.45	49	Figure 4	Basic Structures
StraightSegment	6.7.2.17	103	Figure 28	Layout
String	6.2.2.39	62	Figure 12	Basic Types
TemperatureType	6.1.2.46	50	Figure 3	Basic Structures
TemperatureUnit	6.2.2.40	62	Figure 13	Basic Types
TextAnchorLocation	6.2.2.41	63	Figure 15	Basic Types
TextualAnnotation	6.7.2.18	104	Figure 26	Layout
Time	6.2.2.42	63	Figure 12	Basic Types
Timestamp	6.2.2.43	63	Figure 12	Basic Types
TimeUnit	6.2.2.44	63	Figure 13	Basic Types
TransformationList	6.7.2.19	105	Figure 24	Layout
TransformationOperation	6.7.2.20	105	Figure 24	Layout
Translation	6.7.2.21	106	Figure 24	Layout
UniqueEntity	6.4.2.5	70	Figure 18	Conceptual Framework
UnitDefaults	6.5.2.6	75	Figure 19	Document Definition
UnitTypeName	6.2.2.45	63	Figure 14	Basic Types
UnlimitedCardinality	6.2.2.46	63	Figure 12	Basic Types
URI	6.2.2.47	64	Figure 12	Basic Types
VariableCostData	6.1.2.47	50	Figure 8	Basic Structures
VolumeType	6.1.2.48	51	Figure 3	Basic Structures
VolumeUnit	6.2.2.48	64	Figure 13	Basic Types
WeightType	6.1.2.49	51	Figure 3	Basic Structures
WeightUnit	6.2.2.49	64	Figure 13	Basic Types

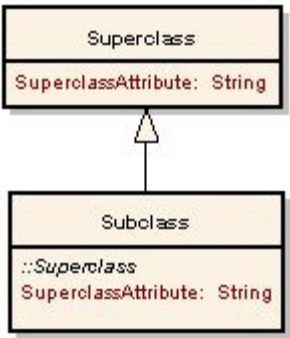
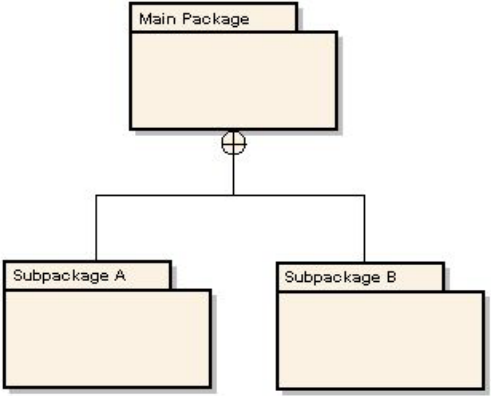
Annex C UML Quick Reference

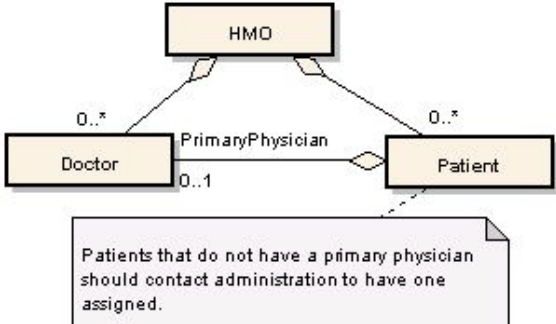

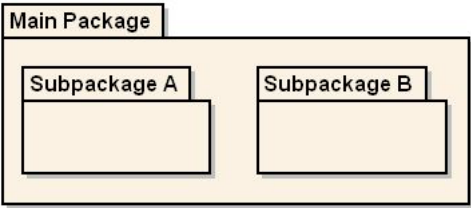
The Unified Modeling Language (UML) defines many types of diagrams and many kinds of modeling constructs that can be used on those diagrams. In the CMSD model, only the package and the class diagram are used. Also, only a small subset of the modeling constructs that are available for use with package and class diagrams are used. In the table below, brief descriptions of the diagrams and the modeling constructs used to define the CMSD model are presented.

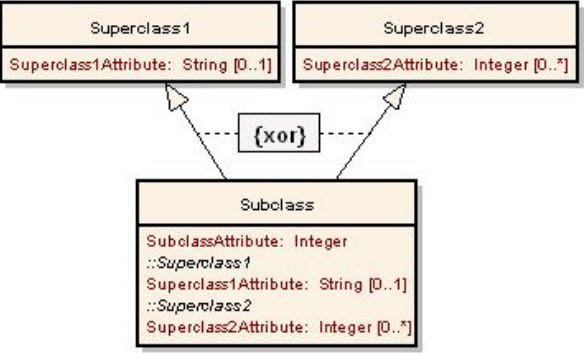
UML Concept Concept Description	Notation Example
<p>abstract class</p> <p>An abstract class provides a means to define a type for a model entity that will never be instantiated in any implementation based on the model. Abstract classes are frequently used in defining generalization/specialization associations between classes that represent general concepts and the classes that specialize them.</p> <p>A class is designated to be abstract if its name is presented in <i>italics</i>.</p>	
<p>aggregation</p> <p>A relationship between two classes where the entity represented by one class is considered to be a part of the entity represented by the other class. The “whole” class is called the aggregating class and the “part” class is called the aggregated class.</p> <p>The relationship is modeled by connecting the aggregated class to the aggregating class with a line that has an open diamond at the end near the aggregating class.</p> <p>At the end of the line near the aggregated class, two additional pieces of information may appear. A multiplicity specification may appear that indicates the range for the number of instances of the aggregated class that may be associated with an instance of the aggregating class. The omission of a multiplicity specification indicates a one to one relationship. In addition, a role name may be specified. The role name indicates the function or purpose of the aggregated class in its relationship with the aggregating class. If the role name is not explicitly specified, the name of the class is the role name.</p>	

UML Concept Concept Description	Notation Example
<p>attribute</p> <p>A characteristic or trait of a class. The attributes of a class are defined by attribute definitions in the attributes compartment of a class.</p> <p>An attribute definition contains:</p> <ol style="list-style-type: none"> 1. the attribute name, followed by a colon (required) 2. the data type of the attribute (required) 3. the multiplicity specification (optional) 4. the default value specification (optional) <p>An attribute of a class is also defined by an aggregation relationship between two classes.</p>	 <pre> classDiagram class SimpleClass { Attribute1: AttributeDataType Attribute2: Integer [0..1] = 0 Attribute3: Decimal [0..*] } </pre> <p>The diagram shows a class box for 'SimpleClass'. The top compartment contains the class name. The bottom compartment contains three attribute definitions: 'Attribute1: AttributeDataType', 'Attribute2: Integer [0..1] = 0', and 'Attribute3: Decimal [0..*]'.</p>
<p>class</p> <p>A type of object or data element that is a part of a UML model. The typical features that are found in a class definition are the class name, attribute definitions, class stereotype, and superclass name. Only the class name is required.</p> <p>In CMSD, only classes that are enumerations have a stereotype. In addition, the superclass feature is only used to indicate that the class being defined has a superclass, and that superclass is defined on another diagram. When both a class and its superclass are defined on the same diagram, an explicit inheritance relationship is shown.</p>	 <pre> classDiagram class StereotypeClass { <<StereotypeName>> ClassName Attribute1Name: AttributeDataTypeA Attribute2Name: AttributeDataTypeB [0..*] } StereotypeClass -- > SuperClassName </pre> <p>The diagram shows a class box for 'StereotypeClass'. The top compartment contains the stereotype «StereotypeName» and the class name 'ClassName'. The bottom compartment contains two attribute definitions: 'Attribute1Name: AttributeDataTypeA' and 'Attribute2Name: AttributeDataTypeB [0..*]'. A dashed line with an open arrow points from the top-right corner of the box to the text 'SuperClassName'.</p>
<p>class diagram</p> <p>A depiction of some of the UML classes that make up a model and the relationships between those classes and other model elements.</p>	 <pre> classDiagram class BaseClass { BaseClassAttribute: Integer } class DerivedClass { DerivedClassAttribute: Decimal <<BaseClass>> BaseClassAttribute: Integer } class RelatedClass BaseClass < -- DerivedClass RelatedClass o-- DerivedClass </pre> <p>The diagram shows three class boxes. 'BaseClass' has one attribute: 'BaseClassAttribute: Integer'. 'DerivedClass' has two attributes: 'DerivedClassAttribute: Decimal' and 'BaseClassAttribute: Integer', with a stereotype «BaseClass» above the second attribute. 'RelatedClass' is connected to 'DerivedClass' by a solid line with an open diamond at the 'RelatedClass' end. An inheritance arrow points from 'DerivedClass' to 'BaseClass'.</p>

UML Concept Concept Description	Notation Example
<p>constraint</p> <p>The specification of a rule or restriction that must be followed for instances of the attached model element or elements.</p> <p>A constraint is constructed by enclosing the text of the rule in curly braces ({}) and placing this information in the body of a note symbol attached to a model element.</p>	 <p>The diagram shows a class named 'Spaceship' with an attribute 'Speed: Decimal'. A note box is attached to the class with the text: '{The value of the Speed attribute should be greater than zero and less than the speed of light.'}</p>
<p>derived association</p> <p>An explicit indication of a relationship between two classes that is only implied by other model elements. A derived association does not semantically add information to a model. Its purpose is to highlight an implicit model relationship to enhance the understandability of the model.</p> <p>A derived association takes the form of a line between two classes that has a role name by one class describing the role or purpose that class plays in the relationship with the other class. The role name always has a slash (/) as a prefix.</p>	 <p>The diagram shows three classes: 'AwardEvent', 'AwardCategory', and 'Person'. 'AwardEvent' has a derived association to 'AwardCategory' with a multiplicity of '1..*'. 'AwardCategory' has a derived association to 'Person' with a multiplicity of '0..*' and the role name 'CategoryNominee'. 'AwardEvent' also has a role name 'AwardNominee' associated with the 'Person' class.</p>
<p>enumeration</p> <p>A class that defines a set of constant string values. The constant string values are referred to as enumeration literals.</p> <p>In the class definition for an enumeration class, an enumeration stereotype («enumeration») always appears over the class name. The enumeration literals appear defined by the class appear on separate lines in the attributes compartment of the class.</p>	 <p>The diagram shows a class named 'Gender' with the stereotype «enumeration». The attributes compartment contains two literals: 'male:' and 'female:'.</p>

UML Concept Concept Description	Notation Example
<p>inheritance</p> <p>A relationship between two classes where one class is considered to be a specialization of another class, or conversely, one class is considered to be a generalization of another class. The general class is often referred to as the base class or superclass, and the specialized class is referred to as the derived class or subclass.</p> <p>The relationship is modeled by connecting the subclass to the superclass with a line that has an open arrow near the superclass.</p> <p>An important aspect of the inheritance relationship is that any attributes defined for the superclass are considered to be defined for the subclass, in addition to any attributes directly defined for the subclass.</p>	 <pre> classDiagram class Superclass { SuperclassAttribute: String } class Subclass { <<Superclass>> SuperclassAttribute: String } Superclass < -- Subclass </pre>
<p>membership notation</p> <p>This notation provides a means to show containment or nesting relationships between packages or classes. The relationship is indicated by connecting the two model elements with a line that has a circle that contains a cross (—⊕) at the end. The end with the circle appears by the model element that contains the other model element.</p>	 <pre> classDiagram package MainPackage { SubpackageA SubpackageB } </pre>

UML Concept Concept Description	Notation Example
<p>multiplicity</p> <p>The range for the number of values that may be associated with an attribute. The multiplicity is provided in a model by including a multiplicity specification either as a part of an attribute definition or a part of an aggregation relationship.</p> <p>A multiplicity specification contains the minimum and maximum number of values that can be associated with attribute, separated by two periods (..). This information is enclosed within square brackets ([]) when it appears as a part of an attribute definition.</p> <p>If the maximum number is unbounded, an asterisk (*) is used instead. If the minimum and maximum numbers are both one (1), the multiplicity specification may be omitted.</p>	<p>[1..5]</p> <p>A multiplicity specification with a range of one to five.</p> <p>[0 .. *]</p> <p>A multiplicity specification with a range of zero to many.</p>
<p>note</p> <p>A note is information of general interest that is attached to a diagram or model element. It is intended to enhance the understandability of the model.</p> <p>The symbol for a note is a rectangular piece of dog-eared paper, connected to a model element by a dashed line.</p>	 <pre> classDiagram class HMO class Doctor class Patient HMO -- "0..*" Doctor HMO -- "0..*" Patient Doctor -- "0..1" Patient : PrimaryPhysician note for Patient "Patients that do not have a primary physician should contact administration to have one assigned." </pre>
<p>package</p> <p>A representation of a system or part of a system being described by a UML model. Within the scope defined by a package, classes and other packages may be defined.</p>	 <pre> classDiagram class PackageName </pre>
<p>package diagram</p> <p>A depiction of some of the UML packages that make up a model and the relationships between those packages and other model elements. In CMSD, package diagrams will only show the nesting relationships between packages in the model.</p>	 <pre> classDiagram class MainPackage class SubpackageA class SubpackageB MainPackage -- SubpackageA MainPackage -- SubpackageB </pre>

UML Concept Concept Description	Notation Example
<p>xor constraint</p> <p>A constraint between two or more relationships associated with a class indicating one and only one of the relationships may be valid in instances of the associated class.</p> <p>When applied to relationships involving multiple inheritance, this constraint indicates that instances of the associated class may only have one of the constrained classes as a superclass.</p> <p>When applied to relationships involving multiple classes being aggregated with another class, this constraint indicates that in instances of the associated class, only one of the aggregation relationships is valid.</p>	 <pre> classDiagram class Superclass1 { Superclass1Attribute: String [0..1] } class Superclass2 { Superclass2Attribute: Integer [0..*] } class Subclass { SubclassAttribute: Integer <<Superclass1>> Superclass1Attribute: String [0..1] <<Superclass2>> Superclass2Attribute: Integer [0..*] } Superclass1 < .. Subclass Superclass2 < .. Subclass Superclass1 Subclass : {xor} Superclass2 Subclass : {xor} </pre> <p>The diagram illustrates a UML class structure where a Subclass inherits from two superclasses, Superclass1 and Superclass2. Superclass1 has an attribute Superclass1Attribute: String [0..1]. Superclass2 has an attribute Superclass2Attribute: Integer [0..*]. The Subclass has its own attribute SubclassAttribute: Integer and includes references to both superclasses, labeled <<Superclass1>> and <<Superclass2>>, with their respective attributes. A dashed box labeled {xor} is positioned between the two inheritance arrows, indicating that only one of the superclasses can be the superclass for any instance of the subclass.</p>