

# Open Geospatial Consortium, Inc.

Date: 2004-02-07

Reference number of this OpenGIS® project document: **OGC 03-105r1**

Version: 3.1.1

Category: OpenGIS® Recommendation Paper

Editors: Simon Cox, Paul Daisey, Ron Lake, Clemens Portele, Arliss Whiteside

## OpenGIS® Geography Markup Language (GML) Implementation Specification

**Notice: This 3.1.1 document is the same as version 3.1.0. The only changes are bug fixes in the online schemas.**

Document type: OpenGIS® Recommendation Paper  
Document subtype: -  
Document stage: Final  
Document language: English

### Copyright notice

Copyright 1999, 2000, 2001, 2002, 2003, 2004 Open GIS Consortium, Inc.

The companies listed above have granted the Open GIS Consortium, Inc. (OGC) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version.

This document does not represent a commitment to implement any portion of this specification in any company's products.

OGC's Legal, IPR and Copyright Statements are found at <http://www.opengis.org/about/?page=ipr&view=ipr>

### NOTICE

Permission to use, copy, and distribute this document in any medium for any purpose and without fee or royalty is hereby granted, provided that you include the above list of copyright holders and the entire text of this NOTICE.

We request that authorship attribution be provided in any software, documents, or other items or products that you create pursuant to the implementation of the contents of this document, or any portion thereof.

No right to create modifications or derivatives of OGC documents is granted pursuant to this license. However, if additional requirements (as documented in the Copyright FAQ at [http://www.opengis.org/about/?page=ipr&view=ipr\\_faq](http://www.opengis.org/about/?page=ipr&view=ipr_faq)) are satisfied, the right to create modifications or derivatives is sometimes granted by the OGC to individuals complying with those requirements.

THIS DOCUMENT IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DOCUMENT ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE DOCUMENT OR THE PERFORMANCE OR IMPLEMENTATION OF THE CONTENTS THEREOF.

The name and trademarks of copyright holders may NOT be used in advertising or publicity pertaining to this document or its contents without specific, written prior permission. Title to copyright in this document will at all times remain with copyright holders.

RESTRICTED RIGHTS LEGEND. Use, duplication, or disclosure by government is subject to restrictions as set forth in subdivision (c)(1)(ii) of the Right in Technical Data and Computer Software Clause at DFARS 252.227.7013

**OpenGIS® is a trademark or registered trademark of Open GIS Consortium, Inc. in the United States and in other countries.**

### Warning

This document is not an OGC Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard.

Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation. Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. Open GIS Consortium shall not be held responsible for identifying any or all such patent rights. However, to date, no such rights have been claimed or identified.

## Contents

1	Scope.....	1
2	Conformance .....	1
3	Normative references .....	2
4	Terms and definitions.....	3
5	Conventions .....	9
5.1	Deprecated parts of previous versions of GML .....	9
5.2	Symbols (and abbreviated terms) .....	9
5.3	UML Notation.....	10
5.4	XML Schema.....	11
6	Overview of GML Schemas.....	11
6.1	Review of GML version 2 – XML for Simple Features .....	11
6.2	GML version 3 – more than Simple Features, plus ISO conformance .....	11
6.3	Backward compatibility .....	12
6.4	GML 3 Application Schemas.....	12
6.5	How GML 3 components are factored into schema documents .....	12
6.6	GML schemas to import into your GML 3 Application Schema .....	14
7	GML schemas – general rules and base schemas .....	15
7.1	Introduction, namespaces, versioning .....	15
7.2	Xlinks – Object Associations and Remote Properties .....	15
7.3	Basic Types .....	15
7.3.1	Overview .....	15
7.3.2	Convenience types .....	16
7.3.2.1	gml:nullType.....	16
7.3.2.2	gml:Null .....	17
7.3.2.3	gml:SignType .....	17
7.3.3	Simple content .....	18
7.3.3.1	gml:booleanOrNull, gml:doubleOrNull, gml:integerOrNull, gml:NameOrNull, gml:stringOrNull .....	18
7.3.3.2	gml:CodeType .....	18
7.3.3.3	gml:MeasureType .....	19
7.3.4	Lists .....	19
7.3.4.1	gml:booleanList, gml:doubleList, gml:integerList, gml:NameList, gml:NCNameList, gml:QNameList, gml:booleanOrNullList, gml:NameOrNullList, gml:doubleOrNullList, gml:integerOrNullList .....	19
7.3.4.2	gml:CodeListType, gml:CodeOrNullListType.....	20
7.3.4.3	gml:MeasureListType, gml:MeasureOrNullListType .....	21
7.4	GML model and syntax.....	21
7.4.1	GML instance documents .....	21
7.4.2	Lexical conventions.....	22
7.4.3	XML schema definition of GML languages.....	22
7.5	gmlBase schema .....	23
7.5.1	Goals of base schema .....	23
7.5.2	Base objects .....	23
7.5.2.1	gml:_Object .....	23
7.5.2.2	gml:_GML, gml:AbstractGMLType .....	23
7.5.3	GML properties.....	24
7.5.3.1	gml:_association, gml:AssociationType .....	24

7.5.3.2	gml:AssociationAttributeGroup.....	25
7.5.4	By value or by reference? .....	25
7.5.4.1	gml:_strictAssociation.....	26
7.5.4.2	gml:_reference, gml:ReferenceType .....	26
7.5.5	Convenience properties and property types.....	27
7.5.5.1	gml:member .....	27
7.5.5.2	gml:stringOrRefType.....	27
7.5.6	Standard properties of GML Objects .....	27
7.5.6.1	gml:metaDataProperty .....	28
7.5.6.2	gml:description.....	28
7.5.6.3	gml:name.....	28
7.5.6.4	gml:StandardObjectProperties .....	28
7.5.6.5	gml:id .....	28
7.5.7	Bags and arrays .....	29
7.5.7.1	gml:ArrayAssociationType.....	29
7.5.7.2	gml:Array, gml:ArrayType, gml:Bag, gml:BagType.....	29
7.5.8	Metadata.....	30
7.5.8.1	gml:metaDataProperty, gml:metaDataPropertyType, gml:_MetaData, gml:AbstractMetaDataType .....	30
7.5.8.2	gml:GenericMetaData, gml:GenericMetaDataType .....	31
8	GML schemas – feature model .....	31
8.1	General Concepts .....	31
8.2	Feature schema.....	31
8.2.1	Features .....	32
8.2.1.1	gml:AbstractFeatureType.....	32
8.2.1.2	gml:_Feature .....	32
8.2.2	Standard feature properties .....	33
8.2.2.1	gml:boundedBy, gml:BoundingShapeType, gml:EnvelopeWithTimePeriod , gml:EnvelopeWithTimePeriodType.....	33
8.2.2.2	gml:location, gml:LocationPropertyType, gml:LocationKeyword, gml:LocationString.....	33
8.2.2.3	gml:priorityLocation, gml:priorityLocationType.....	34
8.2.2.4	gml:featureProperty, gml:FeaturePropertyType, gml:featureMember .....	35
8.2.2.5	gml:featureMembers, gml:FeatureArrayPropertyType .....	35
8.2.3	Geometry Properties.....	35
8.2.4	Topology Properties .....	37
8.2.5	Temporal Properties .....	38
8.2.5.1	General purpose temporal properties .....	38
8.2.5.2	Specific temporal properties .....	38
8.2.6	Defining specific feature types .....	39
8.2.6.1	gml:BoundedFeatureType .....	39
8.2.7	Feature Collections .....	40
8.2.7.1	gml:AbstractFeatureCollectionType .....	40
8.2.7.2	gml:_FeatureCollection, gml:FeatureCollection, gml:FeatureCollectionType .....	40
8.3	Spatial reference system used in a feature or feature collection .....	41
9	GML schemas – basic geometry .....	41
9.1	General Concepts .....	41
9.1.1	Overview .....	41
9.1.2	Abstract Geometry.....	42
9.1.2.1	gml:AbstractGeometryType .....	42
9.1.2.2	SRSReferenceGroup .....	42
9.1.2.3	SRSInformationGroup.....	43
9.1.2.4	gml:_Geometry .....	43
9.1.2.5	gml:Geometry.PropertyType .....	43
9.1.2.6	gml:GeometryArray.PropertyType .....	44
9.1.3	Coordinate Geometry .....	44

9.1.3.1	gml:DirectPositionType, gml:pos .....	44
9.1.3.2	gml:DirectPositionListType, gml:posList .....	44
9.1.3.3	gml:geometricPositionGroup .....	45
9.1.3.4	gml:geometricPositionListGroup .....	45
9.1.3.5	gml:CoordinatesType, gml:coordinates .....	45
9.1.3.6	gml:CoordType, gml:coord .....	46
9.1.4	Vectors .....	46
9.1.4.1	gml:VectorType, gml:Vector .....	46
9.1.5	Envelopes .....	47
9.1.5.1	gml:EnvelopeType, gml:Envelope .....	47
9.2	Geometric Primitives .....	47
9.2.1	Simple Geometric Primitives (0- and 1-dimensional) .....	47
9.2.1.1	gml:AbstractGeometricPrimitiveType, gml:_GeometricPrimitive .....	47
9.2.1.2	gml:PointType, gml:Point .....	48
9.2.1.3	gml:PointPropertyType, gml:pointProperty .....	48
9.2.1.4	gml:PointArrayPropertyType, gml:pointArrayProperty .....	49
9.2.1.5	gml:AbstractCurveType, gml:_Curve .....	49
9.2.1.6	gml:CurvePropertyType, gml:curveProperty .....	49
9.2.1.7	gml:CurveArrayPropertyType, gml:curveArrayProperty .....	50
9.2.1.8	gml:LineStringType, gml:LineString .....	50
9.2.1.9	gml:LineStringPropertyType, gml:lineStringProperty .....	51
9.2.2	Simple Geometric Primitives (2-dimensional) .....	51
9.2.2.1	gml:AbstractSurfaceType, gml:_Surface .....	51
9.2.2.2	gml:SurfacePropertyType, gml:surfaceProperty .....	51
9.2.2.3	gml:SurfaceArrayPropertyType, gml:surfaceArrayProperty .....	52
9.2.2.4	gml:PolygonType, gml:Polygon .....	52
9.2.2.5	gml:exterior, gml:interior, gml:outerBoundaryIs, gml:innerBoundaryIs .....	52
9.2.2.6	gml:AbstractRingType, gml:_Ring .....	53
9.2.2.7	gml:AbstractRingPropertyType .....	53
9.2.2.8	gml:LinearRingType, gml:LinearRing .....	53
9.2.2.9	gml:LinearRingPropertyType .....	54
9.2.2.10	gml:PolygonPropertyType, gml:polygonProperty .....	54
10	GML schemas – more geometric primitives .....	54
10.1	Overview .....	54
10.2	Additional geometric primitives .....	55
10.2.1	1-dimensional geometric primitives .....	55
10.2.1.1	gml:CurveType, gml:Curve .....	55
10.2.1.2	gml:AbstractCurveSegmentType, gml:_CurveSegment .....	55
10.2.1.3	gml:CurveSegmentArrayType, gml:segments .....	56
10.2.1.4	gml:CurveInterpolationType .....	56
10.2.1.5	gml:LineStringSegmentType, gml:LineStringSegment .....	56
10.2.1.6	gml:ArcStringType, gml:ArcString .....	57
10.2.1.7	gml:ArcType, gml:Arc .....	58
10.2.1.8	gml:CircleType, gml:Circle .....	59
10.2.1.9	gml:ArcStringByBulgeType, gml:ArcStringByBulge .....	59
10.2.1.10	gml:ArcByBulgeType, gml:ArcByBulge .....	60
10.2.1.11	gml:ArcByCenterPointType, gml:ArcByCenterPoint .....	61
10.2.1.12	gml:CircleByCenterPointType, gml:CircleByCenterPoint .....	62
10.2.1.13	gml:CubicSplineType, gml:CubicSpline .....	62
10.2.1.14	gml:BSplineType, gml:BSpline .....	64
10.2.1.15	gml:KnotType, gml:KnotPropertyType .....	65
10.2.1.16	gml:KnotTypesType .....	65
10.2.1.17	gml:BezierType, gml:Bezier .....	65
10.2.1.18	gml:OrientableCurveType, gml:OrientableCurve, gml:baseCurve .....	66
10.2.1.19	gml:OffsetCurveType, gml:OffsetCurve .....	67
10.2.1.20	gml:AffinePlacementType, gml:AffinePlacement .....	68
10.2.1.21	gml:ClothoidType, gml:Clothoid .....	68

10.2.1.22	gml:GeodesicStringType, gml:GeodesicString .....	69
10.2.1.23	gml:GeodesicType, gml:Geodesic .....	70
10.2.2	2-dimensional geometric primitives.....	70
10.2.2.1	gml:SurfaceType, gml:Surface .....	70
10.2.2.2	gml:AbstractSurfacePatchType, gml:_SurfacePatch .....	70
10.2.2.3	gml:SurfacePatchArrayType, gml:patches .....	71
10.2.2.4	gml:SurfaceInterpolationType .....	71
10.2.2.5	gml:PolygonPatchType.....	71
10.2.2.6	gml:TriangleType, gml:Triangle .....	72
10.2.2.7	gml:RectangleType, gml:Rectangle .....	72
10.2.2.8	gml:RingType.....	72
10.2.2.9	gml:Ring.PropertyType.....	73
10.2.2.10	gml:OrientableSurfaceType, gml:OrientableSurface, gml:baseSurface .....	73
10.2.2.11	gml:PointGrid.....	74
10.2.2.12	gml:AbstractParametricCurveSurfaceType, gml:_ParametricCurveSurface .....	74
10.2.2.13	gml:AbstractGriddedSurfaceType, gml:_GriddedSurface .....	75
10.2.2.14	gml:ConeType, gml:Cone .....	76
10.2.2.15	gml:CylinderType, gml:Cylinder .....	76
10.2.2.16	gml:SphereType, gml:Sphere .....	77
10.2.2.17	gml:PolyhedralSurfaceType, gml:PolyhedralSurface .....	78
10.2.2.18	gml:PolygonPatchArrayType, gml:PolygonPatches .....	78
10.2.2.19	gml:TriangulatedSurfaceType, gml:TriangulatedSurface .....	78
10.2.2.20	gml:TrianglePatchArrayType, gml:TrianglePatches.....	79
10.2.2.21	gml:TinType, gml:Tin .....	79
10.2.2.22	gml:LineStringSegmentArrayType, gml:LineStringSegmentArrayPropertyType .....	80
10.2.3	3-dimensional geometric primitives.....	80
10.2.3.1	gml:AbstractSolidType, gml:_Solid.....	80
10.2.3.2	gml:Solid.PropertyType, gml:solidProperty .....	81
10.2.3.3	gml:SolidArrayType, gml:solidArrayProperty .....	81
10.2.3.4	gml:SolidType, gml:Solid .....	81
11	GML schemas – geometric complex, geometric composites and geometric aggregates .....	82
11.1	Overview .....	82
11.2	Geometric complex and geometric composites .....	82
11.2.1	Composite Geometries.....	83
11.2.1.1	gml:CompositeCurveType, gml:CompositeCurve .....	83
11.2.1.2	gml:CompositeSurfaceType, gml:CompositeSurface .....	83
11.2.1.3	gml:CompositeSolidType, gml:CompositeSolid .....	84
11.2.2	Geometric Complex .....	84
11.2.2.1	gml:GeometricComplexType, gml:GeometricComplex.....	84
11.2.2.2	gml:GeometricComplex.PropertyType .....	84
11.3	Geometric Aggregates.....	85
11.3.1	Aggregates of unspecified dimensionality.....	85
11.3.1.1	gml:AbstractGeometricAggregateType, gml:_GeometricAggregate .....	85
11.3.1.2	gml:MultiGeometryType, gml:MultiGeometry, gml:geometryMember, gml:geometryMembers .....	85
11.3.1.3	gml:MultiGeometry.PropertyType, gml:multiGeometryProperty .....	86
11.3.2	0-dimensional aggregates.....	86
11.3.2.1	gml:MultiPointType, gml:MultiPoint, gml:pointMember, gml:pointMembers .....	86
11.3.2.2	gml:MultiPoint.PropertyType, gml:multiPointProperty .....	86
11.3.3	1-dimensional aggregates.....	87
11.3.3.1	gml:MultiCurveType, gml:multiCurve, gml:curveMembers .....	87
11.3.3.2	gml:MultiCurve.PropertyType, gml:multiCurveProperty .....	87
11.3.3.3	gml:lineStringMember.....	88
11.3.3.4	gml:MultiLineStringType, gml:MultiLineString .....	88
11.3.3.5	gml:MultiLineString.PropertyType .....	88

11.3.4	2-dimensional aggregates.....	88
11.3.4.1	gml:MultiSurfaceType, gml:MultiSurface, gml:surfaceMember, gml:surfaceMembers .....	88
11.3.4.2	gml:MultiSurfacePropertyType, gml:multiSurfaceProperty.....	89
11.3.4.3	gml:MultiPolygonType, gml:MultiPolygon, gml:polygonMember .....	89
11.3.4.4	gml:MultiPolygonPropertyType .....	90
11.3.5	3-dimensional aggregates.....	90
11.3.5.1	gml:MultiSolidType, gml:MultiSolid, gml:solidMember, gml:solidMembers .....	90
11.3.5.2	gml:MultiSolidPropertyType, gml:multiSolidProperty .....	91
12	GML schemas – Coordinate reference systems schemas .....	91
12.1	Overview .....	91
12.1.1	Introduction .....	91
12.1.2	Coordinates .....	91
12.1.3	Some geodetic concepts .....	92
12.1.4	Non-geodetic coordinate reference systems .....	93
12.1.4.1	Introduction .....	93
12.1.4.2	Geocentric CRS .....	93
12.1.4.3	Temporal CRS.....	93
12.1.4.4	Engineering CRS .....	93
12.1.4.5	Image CRS.....	94
12.1.4.6	Derived CRS.....	94
12.1.5	Important XML elements .....	94
12.2	referenceSystems schema.....	96
12.2.1	Overview .....	96
12.2.2	Identification.....	97
12.2.2.1	gml:IdentifierType .....	97
12.2.2.2	gml:version .....	97
12.2.2.3	gml:remarks .....	97
12.2.2.4	gml:SimpleNameType .....	97
12.2.2.5	gml:scope.....	98
12.2.3	Validity.....	98
12.2.3.1	gml:validArea .....	98
12.2.3.2	gml:ExtentType.....	98
12.2.3.3	gml:boundingBox.....	98
12.2.3.4	gml:boundingPolygon .....	98
12.2.3.5	gml:verticalExtent .....	98
12.2.3.6	gml:temporalExtent.....	99
12.2.4	High-level coordinate reference system .....	99
12.2.4.1	gml:_ReferenceSystem.....	99
12.2.4.2	gml:srsName .....	99
12.2.4.3	gml:srsID .....	99
12.2.4.4	gml:referenceSystemRef .....	100
12.2.4.5	gml:_CRS.....	100
12.2.4.6	gml:crsRef .....	100
12.3	coordinateReferenceSystems schema .....	100
12.3.1	Overview .....	100
12.3.1.1	Introduction.....	100
12.3.1.2	Principal types of coordinate reference systems .....	101
12.3.1.3	Compound coordinate reference systems .....	102
12.3.1.4	Derived coordinate reference systems .....	102
12.3.1.5	Schema introduction.....	103
12.3.2	Abstract coordinate reference systems .....	103
12.3.2.1	gml:_CoordinateReferenceSystem.....	103
12.3.2.2	gml:coordinateReferenceSystemRef .....	103
12.3.2.3	gml:_GeneralDerivedCRS.....	104
12.3.2.4	gml:baseCRS .....	104

12.3.2.5	gml:definedByConversion.....	104
12.3.2.6	gml:generalDerivedCRSRef.....	104
12.3.3	Concrete coordinate reference systems .....	104
12.3.3.1	gml:CompoundCRS .....	104
12.3.3.2	gml:includesCRS.....	105
12.3.3.3	gml:compoundCRSRef .....	105
12.3.3.4	gml:GeographicCRS .....	105
12.3.3.5	gml:usesEllipsoidalCS.....	106
12.3.3.6	gml:usesGeodeticDatum .....	106
12.3.3.7	gml:geographicCRSRef .....	106
12.3.3.8	gml:VerticalCRS .....	106
12.3.3.9	gml:usesVerticalCS.....	106
12.3.3.10	gml:usesVerticalDatum.....	106
12.3.3.11	gml:verticalCRSRef .....	107
12.3.3.12	gml:GeocentricCRS.....	107
12.3.3.13	gml:usesCartesianCS .....	107
12.3.3.14	gml:usesSphericalCS.....	107
12.3.3.15	gml:geocentricCRSRef .....	107
12.3.3.16	gml:ProjectedCRS .....	108
12.3.3.17	gml:projectedCRSRef .....	108
12.3.3.18	gml:DerivedCRS .....	108
12.3.3.19	gml:derivedCRSType .....	109
12.3.3.20	gml:usesCS.....	109
12.3.3.21	gml:derivedCRSRef .....	109
12.3.3.22	gml:EngineeringCRS.....	109
12.3.3.23	gml:usesEngineeringDatum .....	110
12.3.3.24	gml:engineeringCRSRef .....	110
12.3.3.25	gml:ImageCRS .....	110
12.3.3.26	gml:usesObliqueCartesianCS.....	110
12.3.3.27	gml:usesImageDatum .....	111
12.3.3.28	gml:imageCRSRef .....	111
12.3.3.29	gml:TemporalCRS .....	111
12.3.3.30	gml:usesTemporalCS.....	111
12.3.3.31	gml:usesTemporalDatum .....	111
12.3.3.32	gml:temporalCRSRef .....	111
12.4	coordinateSystems schema.....	112
12.4.1	Overview .....	112
12.4.1.1	Introduction .....	112
12.4.1.2	Coordinate system types .....	112
12.4.1.3	Coordinate system axis .....	113
12.4.1.4	Schema introduction .....	114
12.4.2	Coordinate system axes .....	115
12.4.2.1	gml:CoordinateSystemAxis .....	115
12.4.2.2	gml:axisName .....	115
12.4.2.3	gml:axisID.....	115
12.4.2.4	gml:axisAbbrev.....	115
12.4.2.5	gml:axisDirection .....	116
12.4.2.6	gml:uom.....	116
12.4.2.7	gml:coordinateSystemAxisRef .....	116
12.4.3	Abstract coordinate system .....	116
12.4.3.1	gml:_CoordinateSystem .....	116
12.4.3.2	gml:csName .....	117
12.4.3.3	gml:csID.....	117
12.4.3.4	gml:usesAxis .....	117
12.4.3.5	Association to a coordinate system axis.gml:coordinateSystemRef .....	117
12.4.4	Concrete coordinate systems .....	118
12.4.4.1	gml:EllipsoidalCS .....	118
12.4.4.2	gml:ellipsoidalCSRef.....	118

12.4.4.3	gml:CartesianCS.....	118
12.4.4.4	gml:cartesianCSRef .....	118
12.4.4.5	gml:VerticalCS .....	119
12.4.4.6	gml:verticalCSRef.....	119
12.4.4.7	gml:TemporalCS.....	119
12.4.4.8	gml:temporalCSRef.....	119
12.4.4.9	gml:LinearCS .....	120
12.4.4.10	gml:linearCSRef.....	120
12.4.4.11	gml:UserDefinedCS.....	120
12.4.4.12	gml:userDefinedCSRef .....	121
12.4.4.13	gml:SphericalCS.....	121
12.4.4.14	gml:sphericalCSRef .....	121
12.4.4.15	gml:PolarCS .....	121
12.4.4.16	gml:polarCSRef .....	122
12.4.4.17	gml:CylindricalCS.....	122
12.4.4.18	gml:cylindricalCSRef .....	122
12.4.4.19	gml:ObliqueCartesianCS .....	122
12.4.4.20	gml:obliqueCartesianCSRef .....	123
12.5	datums schema .....	123
12.5.1	Overview .....	123
12.5.1.1	Introduction.....	123
12.5.1.2	Geodetic datum .....	123
12.5.1.3	Vertical datum.....	124
12.5.1.4	Image datum .....	124
12.5.1.5	Engineering datum.....	125
12.5.1.6	Temporal datum.....	125
12.5.1.7	Schema introduction.....	125
12.5.2	Abstract datum.....	125
12.5.2.1	gml:_Datum.....	125
12.5.2.2	gml:datumName .....	126
12.5.2.3	gml:datumID.....	126
12.5.2.4	gml:anchorPoint .....	126
12.5.2.5	gml:realizationEpoch .....	127
12.5.2.6	gml:datumRef .....	127
12.5.3	Geodetic datum .....	127
12.5.3.1	gml:GeodeticDatum .....	127
12.5.3.2	gml:usesPrimeMeridian .....	127
12.5.3.3	gml:usesEllipsoid.....	127
12.5.3.4	gml:geodeticDatumRef .....	128
12.5.3.5	gml:Ellipsoid .....	128
12.5.3.6	gml:ellipsoidName.....	129
12.5.3.7	gml:ellipsoidID .....	129
12.5.3.8	gml:semiMajorAxis .....	129
12.5.3.9	gml:ellipsoidRef .....	129
12.5.3.10	gml:secondDefiningParameter .....	129
12.5.3.11	gml:inverseFlattening .....	129
12.5.3.12	gml:semiMinorAxis .....	130
12.5.3.13	gml:isSphere .....	130
12.5.3.14	gml:PrimeMeridian .....	130
12.5.3.15	gml:meridianName .....	130
12.5.3.16	gml:meridianID .....	131
12.5.3.17	gml:greenwichLongitude.....	131
12.5.3.18	gml:primeMeridianRef.....	131
12.5.4	Other concrete datums .....	131
12.5.4.1	gml:EngineeringDatum .....	131
12.5.4.2	gml:engineeringDatumRef.....	131
12.5.4.3	gml:ImageDatum .....	132

12.5.4.4	gml:pixelInCell .....	132
12.5.4.5	gml:imageDatumRef.....	132
12.5.4.6	gml:VerticalDatum.....	132
12.5.4.7	gml:verticalDatumType.....	133
12.5.4.8	gml:verticalDatumRef .....	133
12.5.4.9	gml:TemporalDatum.....	133
12.5.4.10	gml:origin .....	134
12.5.4.11	gml:temporalDatumRef.....	134
12.6	coordinateOperations schema .....	134
12.6.1	Overview .....	134
12.6.1.1	Introduction.....	134
12.6.1.2	Coordinate operation types.....	135
12.6.1.3	Coordinate conversions .....	135
12.6.1.4	Concatenated coordinate operation.....	135
12.6.1.5	Pass-through coordinate operation .....	136
12.6.1.6	Operation method and parameters.....	136
12.6.1.7	Schema introduction .....	137
12.6.2	Abstract coordinate operations.....	137
12.6.2.1	gml:_CoordinateOperation .....	137
12.6.2.2	gml:coordinateOperationName.....	138
12.6.2.3	gml:coordinateOperationID .....	138
12.6.2.4	gml:operationVersion .....	138
12.6.2.5	gml:sourceCRS.....	138
12.6.2.6	gml:targetCRS .....	138
12.6.2.7	gml:coordinateOperationRef.....	138
12.6.2.8	gml:_SingleOperation .....	139
12.6.2.9	gml:singleOperationRef.....	139
12.6.2.10	gml:_Operation .....	139
12.6.2.11	gml:operationRef .....	140
12.6.2.12	gml:_GeneralConversion .....	140
12.6.2.13	gml:generalConversionRef.....	140
12.6.2.14	gml:_GeneralTransformation .....	141
12.6.2.15	gml:generalTransformationRef .....	141
12.6.3	Concrete coordinate operations.....	142
12.6.3.1	gml:ConcatenatedOperation .....	142
12.6.3.2	gml:usesSingleOperation .....	142
12.6.3.3	gml:concatenatedOperationRef .....	142
12.6.3.4	gml:PassThroughOperation .....	142
12.6.3.5	gml:modifiedCoordinate .....	143
12.6.3.6	A positive integer defining a position in a coordinate tuple.gml:usesOperation .....	143
12.6.3.7	gml:passThroughOperationRef .....	143
12.6.3.8	gml:Conversion .....	143
12.6.3.9	gml:usesMethod .....	144
12.6.3.10	gml:usesValue .....	144
12.6.3.11	Composition association to a parameter value used by this coordinate operation.gml:conversionRef .....	144
12.6.3.12	gml:Transformation .....	144
12.6.3.13	gml:transformationRef .....	145
12.6.4	Parameter values and groups.....	145
12.6.4.1	gml:_generalParameterValue .....	145
12.6.4.2	gml:parameterValue .....	145
12.6.4.3	gml:value .....	146
12.6.4.4	gml:dmsAngleValue .....	146
12.6.4.5	gml:stringValue .....	146
12.6.4.6	gml:integerValue .....	146
12.6.4.7	gml:booleanValue.....	146
12.6.4.8	gml:valueList .....	146

12.6.4.9	gml:integerValueList .....	146
12.6.4.10	gml:valueFile.....	147
12.6.4.11	gml:valueOfParameter .....	147
12.6.4.12	gml:parameterValueGroup .....	147
12.6.4.13	gml:includesValue.....	147
12.6.4.14	gml:valuesOfGroup .....	147
12.6.5	Operation method .....	148
12.6.5.1	gml:OperationMethod .....	148
12.6.5.2	gml:methodName .....	148
12.6.5.3	gml:methodID .....	148
12.6.5.4	gml:methodFormula.....	148
12.6.5.5	gml:sourceDimensions.....	149
12.6.5.6	gml:targetDimensions.....	149
12.6.5.7	gml:usesParameter .....	149
12.6.5.8	gml:operationMethodRef .....	149
12.6.6	Operation parameters and groups .....	149
12.6.6.1	gml:GeneralOperationParameter .....	149
12.6.6.2	gml:minimumOccurs.....	150
12.6.6.3	gml:abstractGeneralOperationParameterRef .....	150
12.6.6.4	gml:OperationParameter .....	150
12.6.6.5	gml:parameterName.....	151
12.6.6.6	gml:parameterID .....	151
12.6.6.7	gml:operationParameterRef .....	151
12.6.6.8	gml:OperationParameterGroup .....	151
12.6.6.9	gml:groupName .....	152
12.6.6.10	gml:groupID .....	152
12.6.6.11	gml:maximumOccurs .....	152
12.6.6.12	gml:includesParameter .....	152
12.6.6.13	gml:operationParameterGroupRef .....	152
12.7	dataQuality schema .....	153
12.7.1	Overview .....	153
12.7.2	All elements .....	153
12.7.2.1	gml:_positionalAccuracy.....	153
12.7.2.2	gml:measureDescription .....	154
12.7.2.3	gml:absoluteExternalPositionalAccuracy .....	154
12.7.2.4	gml:relativeInternalPositionalAccuracy.....	154
12.7.2.5	gml:result .....	154
12.7.2.6	gml:covarianceMatrix.....	154
12.7.2.7	gml:includesElement .....	155
12.7.2.8	gml:rowIndex .....	155
12.7.2.9	gml:columnIndex.....	155
12.7.2.10	gml:covariance .....	155
13	GML schemas – topology.....	155
13.1	Introduction .....	155
13.2	Topology Model.....	156
13.3	Types and elements .....	156
13.3.1	Schema Includes .....	156
13.3.2	gml:AbstractTopologyType, gml:_Topology .....	156
13.3.3	gml:AbstractTopoPrimitive, gml:_TopoPrimitive .....	156
13.3.4	gml:NodeType, gml:Node.....	157
13.3.5	gml:DirectedNodePropertyType, gml:directedNode .....	157
13.3.6	gml:EdgeType, gml:Edge .....	157
13.3.7	gml:DirectedEdgePropertyType, gml:directedEdge .....	158
13.3.8	gml:FaceType, gml:Face .....	158
13.3.9	gml:DirectedFacePropertyType, gml:directedFace.....	159
13.3.10	gml:TopoSolidType, gml:TopoSolid .....	159

13.3.11	gml:DirectedTopoSolidPropertyType, gml:directedTopoSolid.....	159
13.3.12	gml:isolated .....	160
13.3.13	gml:container .....	160
13.3.14	gml:TopoPointType, gml:TopoPoint .....	161
13.3.15	gml:TopoPointPropertyType, gml:topoPointProperty .....	161
13.3.16	gml:TopoCurveType, gml:TopoCurve .....	161
13.3.17	gml:TopoCurvePropertyType, gml:topoCurveProperty .....	162
13.3.18	gml:TopoSurfaceType, gml:TopoSurface .....	162
13.3.19	gml:TopoSurfacePropertyType, gml:topoSurfaceProperty .....	162
13.3.20	gml:TopoVolumeType, gml:TopoVolume .....	162
13.3.21	gml:TopoVolumePropertyType, gml:topoVolumeProperty .....	163
13.3.22	gml:TopoComplexType, gml:TopoComplex .....	163
13.3.23	Maximal, sub- and super-complexes .....	163
13.3.24	gml:topoPrimitiveMember .....	164
13.3.25	gml:topoPrimitiveMembers .....	165
13.3.26	gml:TopoComplexProperty, gml:topoComplexProperty .....	165
14	GML schemas – temporal information and dynamic features .....	165
14.1	Overview .....	165
14.2	Temporal schema.....	166
14.2.1	Overview .....	166
14.2.2	urn:opengis:specification:gml:schema-xsd:temporal:v3.1.0Temporal objects and properties.....	166
14.2.2.1	gml:_TimeObject .....	166
14.2.2.2	gml:_TimePrimitive .....	166
14.2.2.3	gml:TimePrimitivePropertyType, gml:validTime .....	167
14.2.2.4	gml:RelatedTimeType .....	167
14.2.2.5	gml:_TimeComplex .....	167
14.2.3	Temporal geometry .....	168
14.2.3.1	gml:_TimeGeometricPrimitive .....	168
14.2.3.2	gml:TimelInstant .....	168
14.2.3.3	gml:TimelInstantPropertyType .....	169
14.2.3.4	gml:TimePeriod .....	169
14.2.3.5	gml:timePeriodPropertyType .....	170
14.2.3.6	gml:TimePositionType, gml:timePosition .....	170
14.2.3.7	gml:TimeLengthType, gml:_timeLength, gml:duration, gml:timelInterval, gml:TimeUnitType .....	172
14.3	Temporal topology schema .....	173
14.3.1	Overview .....	173
14.3.2	Temporal topology objects .....	174
14.3.2.1	gml:_TimeTopologyPrimitive .....	174
14.3.2.2	gml:TimeTopologyPrimitivePropertyType .....	175
14.3.2.3	gml:TimeTopologyComplex .....	175
14.3.2.4	gml:TimeNode .....	175
14.3.2.5	gml:TimeNodePropertyType .....	176
14.3.2.6	gml:TimeEdge .....	176
14.3.2.7	gml:TimeEdgePropertyType .....	176
14.3.2.8	gml:SuccessionType .....	176
14.4	Temporal reference systems .....	177
14.4.1	Overview .....	177
14.4.2	Basic temporal reference system.....	177
14.4.2.1	gml:TimeReferenceSystem .....	177
14.4.3	Time Coordinate Systems .....	178
14.4.3.1	gml:TimeCoordinateSystem .....	178
14.4.4	Calendars and clocks .....	179
14.4.4.1	gml:TimeCalendar, gml:TimeCalendarEra .....	179
14.4.4.2	gml:TimeClock .....	180
14.4.5	Time Ordinal Reference Systems .....	181
14.4.5.1	gml:TimeOrdinalReferenceSystem, gml:TimeOrdinalEra .....	181

14.5	Representing dynamic features .....	183
14.5.1	Overview .....	183
14.5.2	gml:dataSource .....	184
14.5.3	Dynamic Properties .....	184
14.5.4	Dynamic Features .....	184
14.5.5	gml:DynamicFeatureCollection .....	184
14.5.6	gml:_TimeSlice .....	185
14.5.7	gml:MovingObjectStatus.....	186
14.5.8	gml:history.....	186
14.5.9	gml:track .....	187
15	GML schemas – definitions and dictionaries.....	187
15.1	Overview .....	187
15.2	Dictionary schema .....	188
15.2.1	gml:Definition, gml:DefinitionType .....	188
15.2.2	gml:Dictionary, gml:DefinitionCollection, gml:DictionaryType .....	189
15.2.3	gml:dictionaryEntry, gml:definitionMember, gml:DictionaryEntryType .....	189
15.2.4	gml:indirectEntry, gml:IndirectEntryType, gml:DefinitionProxy, gml:DefinitionProxyType.....	189
15.2.5	Using Definitions and Dictionaries .....	191
16	GML schemas – units, measures and values.....	191
16.1	Introduction .....	191
16.2	Units schema.....	192
16.2.1	Using Unit Definitions.....	192
16.2.2	gml:unitOfMeasure, gml:UnitOfMeasureType.....	192
16.2.3	gml:UnitDefinition, gml:UnitDefinitionType .....	193
16.2.4	gml: quantityType .....	193
16.2.5	gml:catalogSymbol .....	193
16.2.6	gml:BaseUnit, gml:BaseUnitType, gml:unitsSystem.....	194
16.2.7	gml:DerivedUnit, gml:DerivedUnitType .....	194
16.2.8	gml:derivationUnitTerms, gml:DerivationUnitTermType.....	194
16.2.9	gml:ConventionalUnit, gml:ConventionalUnitType .....	195
16.2.10	gml:conversionToPreferredUnit, gml:roughConversionToPreferredUnit, gml:ConversionToPreferredUnitType, gml:FormulaType .....	195
16.2.11	Example of units dictionary .....	196
16.3	Measures schema .....	202
16.3.1	gml:measure .....	202
16.3.2	Scalar measure types .....	202
16.3.3	gml:angle .....	203
16.3.4	gml:dmsAngle .....	203
16.3.5	gml:degrees .....	204
16.3.6	gml:decimalMinutes.....	205
16.3.7	gml:minutes .....	205
16.3.8	gml:seconds .....	205
16.3.9	gml:AngleChoiceType .....	205
16.4	Value Objects schema .....	205
16.4.1	Introduction .....	205
16.4.2	Value element hierarchy.....	206
16.4.3	gml:Boolean, gml:BooleanList .....	206
16.4.4	gml:Category, gml:CategoryList .....	207
16.4.5	gml:Count, gml:CountList.....	207
16.4.6	gml:Quantity, gml:QuantityList .....	207
16.4.7	gml:_Value, gml:_ScalarValue, gml:_ScalarValueList .....	208
16.4.8	gml:Value .....	208
16.4.9	gml:valueProperty, gml:valueComponent, gml:valueComponents .....	208
16.4.10	gml:CompositeValue .....	209
16.4.11	gml:ValueArray.....	210

16.4.12	Typed ValueExtents: <i>gml:CategoryExtent</i> , <i>gml:CountExtent</i> , <i>gml:QuantityExtent</i> .....	211
16.4.13	<i>gml:BooleanPropertyType</i> , <i>gml:CategoryPropertyType</i> , <i>gml:CountPropertyType</i> , <i>gml:QuantityPropertyType</i> .....	212
17	GML schemas – directions.....	212
17.1	Direction schema .....	212
17.1.1	<i>gml:direction</i> , <i>gml:Direction.PropertyType</i> .....	212
17.1.2	<i>gml:DirectionVector</i> , <i>gml:DirectionVectorType</i> .....	213
17.1.3	<i>gml:CompassPoint</i> , <i>gml:CompassPointEnumeration</i> .....	213
17.1.4	Text Based Directions: <i>gml:DirectionKeyword</i> , <i>gml:DirectionString</i> .....	214
18	GML schemas – observations .....	214
18.1	Observations .....	214
18.2	Observation schema.....	214
18.2.1	<i>gml:Observation</i> .....	214
18.2.2	<i>gml:using</i> .....	215
18.2.3	<i>gml:target</i> .....	215
18.2.4	<i>gml:resultOf</i> .....	216
18.3	Examples.....	216
18.3.1	<i>gml:Observation</i> .....	216
18.3.2	<i>gml:DirectedObservation</i> .....	216
18.3.3	<i>gml:DirectedObservationAtDistance</i> .....	217
19	GML schemas – coverages .....	218
19.1	The coverage model and representations.....	218
19.1.1	General Remarks.....	218
19.1.2	Formal description of a coverage .....	218
19.1.3	Coverage in GML.....	219
19.2	Grids schema .....	220
19.2.1	Unrectified Grid ( <i>gml:Grid</i> ).....	220
19.2.2	Rectified Grid ( <i>gml:RectifiedGrid</i> ).....	221
19.3	Coverage schema .....	222
19.3.1	<i>gml:AbstractCoverageType</i> , <i>gml:_Coverage</i> .....	222
19.3.2	<i>gml:AbstractDiscreteCoverageType</i> , <i>gml:_DiscreteCoverage</i> .....	223
19.3.3	<i>gml:AbstractContinuousCoverageType</i> , <i>gml:_ContinuousCoverage</i> .....	223
19.3.4	<i>gml:domainSet</i> , <i>gml:DomainSetType</i> .....	224
19.3.5	<i>gml:rangeSet</i> , <i>gml:RangeSetType</i> .....	224
19.3.6	<i>gml:DataBlock</i> .....	225
19.3.7	<i>gml:rangeParameters</i> .....	225
19.3.8	<i>gml:tupleList</i> .....	225
19.3.9	<i>gml:doubleOrNullTupleList</i> .....	226
19.3.10	<i>gml:File</i> , <i>gml:FileType</i> .....	226
19.3.11	<i>gml:coverageFunction</i> , <i>gml:CovarianceFunctionType</i> .....	228
19.3.12	<i>gml:MappingRule</i> .....	228
19.3.13	<i>gml:GridFunction</i> , <i>gml:GridFunctionType</i> .....	228
19.3.14	<i>gml:sequenceRule</i> , <i>gml:SequenceRuleType</i> , <i>gml:SequenceRuleNames</i> .....	229
19.3.15	Specific Coverage Types in GML 3.1 .....	231
19.3.16	MultiPoint Coverage .....	231
19.3.17	MultiCurve Coverage .....	233
19.3.18	MultiSurface Coverage .....	234
19.3.19	MultiSolid Coverage.....	236
19.3.20	Gridded Coverage .....	236
19.3.21	RectifiedGrid Coverage .....	237
20	GML schemas – default styling .....	239
20.1	General concepts .....	239
20.2	Top-level styling elements .....	241
20.2.1	<i>gml:defaultStyle</i> .....	241
20.2.2	<i>gml:Style</i> .....	242

20.3	Feature style .....	243
20.3.1	gml:FeatureStyle .....	243
20.3.2	featureType .....	243
20.3.3	baseType .....	244
20.3.4	featureConstraint .....	244
20.3.5	queryGrammar .....	244
20.4	Geometry Style .....	244
20.5	Topology Style .....	245
20.6	Label Style .....	246
20.7	Common styling elements .....	247
20.7.1	Overview .....	247
20.7.2	gml:symbol .....	247
20.7.3	gml:styleVariation .....	248
20.7.4	gml:spatialResolution .....	249
20.7.5	animation .....	249
20.8	Graph Style .....	249
21	Modularisation and Dependencies .....	250
22	Profiles .....	253
22.1	Profiles of GML and application schemas .....	253
22.2	Definition of Profile .....	253
22.3	Relation to application schema .....	253
22.4	Rules for elements and types in a profile .....	253
22.5	Recommendations for application schemas using GML profiles .....	254
22.6	Summary of Rules for GML profiles .....	255
23	Rules for Application Schemes .....	255
23.1	GML Documents .....	255
23.2	GML Application Schemas .....	256
23.2.1	Target Namespace .....	257
23.2.2	Import GML Schema .....	257
23.2.3	Object Type Derivation .....	257
23.2.4	Elements Representing Objects .....	257
23.2.5	Property Type Derivation .....	257
23.2.6	Elements Representing Properties .....	258
23.3	Schemas defining Features and Feature Collections .....	258
23.3.1	Target Namespace .....	258
23.3.2	Import feature.xsd .....	258
23.3.3	Feature Type Derivation .....	259
23.3.4	Elements Representing Features .....	259
23.3.5	Application Features are Features .....	259
23.3.6	GML Feature Collection Document .....	259
23.4	Schemas defining Geometries .....	259
23.4.1	Target Namespace .....	259
23.4.2	Import a GML geometry schema .....	259
23.4.3	User-defined Geometry Types and Geometry Property Types .....	260
23.4.3.1	User-defined Geometry Types .....	260
23.4.3.2	User-defined Geometry Property Types .....	260
23.5	Schemas defining Topologies .....	261
23.5.1	Target Namespace .....	261
23.5.2	Import topology.xsd .....	261
23.5.3	User-defined Topology Types and Topology Property Types .....	261
23.5.3.1	User-defined Topology Types .....	261
23.5.3.2	User-defined Topology Property Types .....	261
23.6	Schemas defining Time .....	261
23.6.1	Target Namespace .....	261
23.6.2	Import temporal.xsd .....	261

23.6.3 User-defined Temporal Types and Temporal Property Types .....	262
23.6.3.1 User-defined Temporal Types.....	262
23.6.3.2 User-defined Temporal Property Types.....	262
23.7 Schemas defining Coordinate Reference Systems.....	262
23.7.1 Target Namespace .....	263
23.7.2 Import coordinateReferenceSystem.xsd .....	263
23.8 Schemas defining Coverages .....	263
23.8.1 Target Namespace .....	263
23.8.2 Import coverage.xsd .....	263
23.8.3 Coverages shall derive from gml:AbstractDiscreteCoverageType or gml:AbstractContinuousCoverageType .....	263
23.8.4 Range Parameters shall be Derived from gml:ValueType .....	264
23.8.5 Coverage Document .....	265
23.9 Schemas defining Observations .....	265
23.9.1 Target Namespace .....	265
23.9.2 Import observation.xsd .....	265
23.9.3 Observations shall derive from gml:ObservationType .....	265
23.9.4 Observation Collections shall derive from gml:CollectionType .....	265
23.9.5 Observations are Features.....	265
23.9.6 Observation Collection Document .....	266
23.10 Schemas defining Dictionaries and Definitions .....	266
23.10.1 Target Namespace .....	266
23.10.2 Import dictionary.xsd.....	266
23.10.3 Definitions shall derive from gml:DefinitionType.....	266
23.10.4 Dictionaries shall derive from gml:DictionaryType.....	267
23.11 Schemas defining Values.....	267
23.11.1 Target Namespace .....	267
23.11.2 Import valueObjects.xsd or basicTypes.xsd.....	267
23.11.3 Construction of New Value Types.....	267
Annex A (normative) Abstract Test Suite.....	268
A.1 Conformance Class A. Conformance of the XML Implementation of GML data.....	268
A.1.1 Existence of an applicable GML application schema.....	268
A.1.2 Conformance of the applicable application schema .....	268
A.1.3 Conformance of the data set.....	268
A.2 Conformance Class B. Conformance of the XML Implementation of GML Application Schema.....	268
A.2.1 XMLSchema valid .....	268
A.2.2 GML model valid .....	269
A.3 Conformance Class C. Conformance of the GML Interface Implementation for GML Data.....	269
A.3.1 Serialization capability test .....	269
A.3.2 Serialization validity test.....	269
A.3.3 Transforming to and from XML format.....	269
A.4 Conformance Class D. Conformance of the GML Interface Implementation for GML Application Schema.....	270
A.4.1 Serialization capability test .....	270
A.4.2 Serialization validity test.....	270
A.4.3 Transforming to and from XML format.....	270
Annex B (normative) Conformance.....	271
B.1 GML Object Rules.....	271
B.1.1 GML Objects .....	271
B.1.2 GML Properties.....	271
B.2 Application Schemas .....	271
B.3 Conformance Classes .....	271
B.3.1 Conformance Requirements .....	271
Annex C (normative) GML Schemas .....	273

C.1	<b>basicTypes.xsd</b> .....	273
C.2	<b>coordinateOperations.xsd</b> .....	277
C.3	<b>coordinateReferenceSystems.xsd</b> .....	291
C.4	<b>coordinateSystems.xsd</b> .....	299
C.5	<b>coverage.xsd</b> .....	307
C.6	<b>dataQuality.xsd</b> .....	314
C.7	<b>datums.xsd</b> .....	316
C.8	<b>defaultStyle.xsd</b> .....	324
C.9	<b>dictionary.xsd</b> .....	331
C.10	<b>direction.xsd</b> .....	333
C.11	<b>dynamicFeature.xsd</b> .....	334
C.12	<b>feature.xsd</b> .....	336
C.13	<b>geometryAggregates.xsd</b> .....	339
C.14	<b>geometryBasic0d1d.xsd</b> .....	346
C.15	<b>geometryBasic2d.xsd</b> .....	355
C.16	<b>geometryComplexes.xsd</b> .....	359
C.17	<b>geometryPrimitives.xsd</b> .....	362
C.18	<b>gml.xsd</b> .....	389
C.19	<b>gmlBase.xsd</b> .....	389
C.20	<b>grids.xsd</b> .....	394
C.21	<b>measures.xsd</b> .....	395
C.22	<b>observation.xsd</b> .....	398
C.23	<b>referenceSystems.xsd</b> .....	400
C.24	<b>temporal.xsd</b> .....	404
C.25	<b>temporalReferenceSystems.xsd</b> .....	409
C.26	<b>temporalTopology.xsd</b> .....	413
C.27	<b>topology.xsd</b> .....	416
C.28	<b>units.xsd</b> .....	423
C.29	<b>valueObjects.xsd</b> .....	426
C.30	<b>xlink.xsd</b> .....	431
	<b>Annex D (informative) ISO 19100 Profile and Application Schema of GML</b> .....	434
D.1	<b>General Approach</b> .....	434
D.2	<b>Profile of the ISO 19100 Harmonized Model used by GML</b> .....	434
D.2.1	<b>ISO 19103 Conceptual Schema Language</b> .....	435
D.2.2	<b>ISO 19107 Spatial Schema (Geometry)</b> .....	436
D.2.2.1	<b>Overview</b> .....	436
D.2.2.2	<b>Geometry root</b> .....	437
D.2.2.3	<b>Geometry primitive</b> .....	439
D.2.2.4	<b>Coordinate Geometry</b> .....	445
D.2.2.5	<b>Geometry aggregates</b> .....	453
D.2.2.6	<b>Geometry complex</b> .....	454
D.2.2.7	<b>Conformance</b> .....	456
D.2.3	<b>ISO 19107 Spatial Schema (Topology)</b> .....	457
D.2.3.1	<b>Overview</b> .....	457
D.2.3.2	<b>Topology root</b> .....	457
D.2.3.3	<b>Topology primitive</b> .....	459
D.2.3.4	<b>Topology complex</b> .....	463
D.2.3.5	<b>Conformance</b> .....	463
D.2.4	<b>ISO 19108 Temporal Schema</b> .....	464
D.2.4.1	<b>Overview</b> .....	464
D.2.4.2	<b>Temporal Objects</b> .....	465
D.2.4.3	<b>Concrete Temporal Geometric Primitives</b> .....	466
D.2.4.4	<b>Temporal Duration</b> .....	467
D.2.4.5	<b>Temporal Position</b> .....	468
D.2.4.6	<b>Temporal Topology</b> .....	469
D.2.4.7	<b>Temporal Reference Systems</b> .....	469
D.2.4.8	<b>Calendars and Clocks</b> .....	470

D.2.4.9	Time Coordinate Systems .....	471
D.2.4.10	Temporal Ordinal Reference System .....	471
D.2.4.11	Conformance .....	471
D.2.5	ISO 19109 Rules for App Schema .....	471
D.2.6	ISO 19111 Spatial Referencing By Coordinates.....	472
D.2.7	ISO 19112 Spatial Referencing By Geographic Identifiers .....	474
D.2.8	ISO 19115 Metadata .....	475
D.2.9	ISO 19117 Portrayal .....	475
D.2.9.1	Overview.....	475
D.2.9.2	Portrayal Mechanism from ISO 19117 .....	475
D.2.9.3	Portrayal Service .....	479
D.2.9.4	Portrayal Catalogue.....	479
D.2.9.5	PF_PortrayalCatalogue .....	480
D.2.9.6	PF_FeaturePortrayal .....	481
D.2.9.7	PF_PortrayalRule.....	481
D.2.9.8	Portrayal Specification.....	481
D.2.9.9	PF_PortrayalSpecification.....	482
D.2.9.10	PF_PortrayalOperation .....	482
D.2.9.11	Portrayal of textual data.....	482
D.2.9.12	Representation of symbols .....	482
D.2.9.13	Tabular Comparison.....	483
D.2.9.14	Conformance .....	483
D.2.10	ISO 19123 Coverages.....	485
D.3	Extension of the profile of the ISO 19100 Harmonized Model used by GML .....	487
D.3.1	Overview .....	487
D.3.2	Package “basicTypes” .....	488
D.3.3	Package “gmlBase” .....	489
D.3.4	Package “dictionary” .....	489
D.3.5	Package “units” .....	490
D.3.6	Package “measures” .....	490
D.3.7	Package “geometryBasic0d01”.....	492
D.3.8	Package “geometryBasic2d” .....	492
D.3.9	Package “geometryPrimitives” .....	493
D.3.10	Package “geometryAggregates” .....	494
D.3.11	Package “geometryComplexes” .....	494
D.3.12	Package “topology” .....	495
D.3.13	Package “direction” .....	496
D.3.14	Package “valueObjects” .....	496
D.3.15	Package “temporal” .....	498
D.3.16	Package “dynamicFeature” .....	499
D.3.17	Package “feature” .....	500
D.3.18	Package “observation”.....	500
D.3.19	Package “grids” .....	500
D.3.20	Package “coverage” .....	501
D.3.21	Packages “coordinateOperations”, “coordinateReferenceSystems”, “coordinateSystems”, “dataQuality”, “datums”, “referenceSystems”.....	501
D.3.22	Package “defaultStyle”.....	501
	<b>Annex E (normative) UML-to-GML Application Schema Encoding Rules .....</b>	<b>502</b>
E.1	General concepts .....	502
E.2	Encoding Rules .....	502
E.2.1	General encoding requirements .....	502
E.2.1.1	Application schema.....	502
E.2.1.2	Character repertoire and languages.....	508
E.2.1.3	Exchange metadata.....	508
E.2.1.4	Dataset and object identification .....	508
E.2.1.5	Update mechanism.....	508
E.2.2	Input data structure .....	508
E.2.3	Output data structure .....	509

<b>E.2.4 Conversion rules</b> .....	509
E.2.4.1 General concepts .....	509
E.2.4.2 UML Packages .....	510
E.2.4.3 UML Classes (general rules) .....	510
E.2.4.4 UML Classes (basic types) .....	510
E.2.4.4 UML Classes (data types) .....	512
E.2.4.5 UML Classes (feature types) .....	513
E.2.4.6 UML Classes (object types).....	513
E.2.4.7 UML Classes (enumerations).....	514
E.2.4.8 UML Classes (code lists) .....	514
E.2.4.9 UML Classes (unions) .....	515
E.2.4.10 UML Attributes and Association roles .....	515
E.2.4.11 Documentation .....	517
E.2.4.12 Imported Classes from the ISO 19100 Harmonized Model.....	517
<b>E.3 Example &lt;informative&gt;</b> .....	519
<b>Annex F (normative) GML-to-UML Application Schema Encoding Rules</b> .....	522
<b>F.1 General concepts</b> .....	522
<b>F.2 Encoding Rules</b> .....	522
<b>F.2.1 General encoding requirements</b> .....	522
F.2.1.1 General remarks .....	522
F.2.1.2 GML schema .....	523
F.2.1.3 Character repertoire and languages.....	527
F.2.1.4 Exchange metadata .....	528
F.2.1.5 Dataset and object identification .....	528
F.2.1.6 Update mechanism.....	528
F.2.1.7 Input data structure.....	528
<b>F.2.2 Output data structure</b> .....	528
<b>F.2.3 Conversion rules</b> .....	528
F.2.3.1 General concepts .....	528
F.2.3.2 GML Schema Documents .....	530
F.2.3.3 GML Object Types .....	530
F.2.3.4 GML Object Types (imported from the GML schemas).....	531
F.2.3.5 Basic Types.....	531
F.2.3.6 GML Data Types .....	532
F.2.3.7 Enumerations.....	532
F.2.3.8 Code lists.....	533
F.2.3.9 GML properties .....	533
F.2.3.10 Documentation .....	533
<b>Annex G (Informative) Guidelines for Subsetting GML Schemas</b> .....	534
<b>G.1 General</b> .....	534
<b>G.2 depends.xslt</b> .....	535
<b>G.3 gmlSubset.xslt</b> .....	540
<b>G.4 utility.xslt</b> .....	543

## Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. Open GIS Consortium Inc. and ISO shall not be held responsible for identifying any or all such patent rights.

International Standard ISO 19136 was prepared by the Technical Committee of the Open GIS Consortium and Technical Committee ISO/TC 211, *Geographic information/Geomatics*.

This edition of GML supersedes version 3.0.0 (OGC document 02-023r4).

GML is a living standard, originally developed within the Open GIS Consortium (OGC). This ISO/TC 211 Committee Draft is targeted to become GML version 3.1 within OGC. As GML is already used in several commercial data and software products worldwide, compatibility of this version with previous versions of GML is important. OGC has decided that the best way to introduce changes to existing structures within GML is to deprecate elements, types, attributes and groups that shall not be used in new applications. These deprecated structures will then be removed from GML in a future version – depending on the adoption of these changes by the market.

Member bodies and liaisons are invited to address in their comments the issue whether this International Standard should contain the deprecated structures, too, or if they should be removed from the standard. In the latter case, ISO 19136 would be a profile of the GML Implementation Specification according to the rules of this standard.

## Introduction

Geography Markup Language is an XML grammar written in XML Schema for the modelling, transport, and storage of geographic information.

The key concepts used by Geography Markup Language (GML) to model the world are drawn from the OpenGIS® Abstract Specification and the ISO 19100 series.

GML provides a variety of kinds of objects for describing geography including features, coordinate reference systems, geometry, topology, time, units of measure and generalized values.

A geographic feature is "an abstraction of a real world phenomenon; it is a geographic feature if it is associated with a location relative to the Earth". So a digital representation of the real world can be thought of as a set of features. The state of a feature is defined by a set of properties, where each property can be thought of as a {name, type, value} triple.

The number of properties a feature may have, together with their names and types, are determined by its type definition. Geographic features with geometry are those with properties that may be geometry-valued. A feature collection is a collection of features that can itself be regarded as a feature; as a consequence a feature collection has a feature type and thus may have distinct properties of its own, in addition to the features it contains.

Geographic features in GML include coverages and observations as subtypes.

A coverage is a sub-type of feature that has a coverage function with a spatial domain and a value set range of homogeneous 2 to n dimensional tuples. A coverage can represent one feature or a collection of features "to model and make visible spatial relationships between, and the spatial distribution of, earth phenomena."

An observation models the act of observing, often with a camera, a person or some form of instrument ("an act of recognizing and noting a fact or occurrence often involving measurement with instruments"). An observation is considered to be a GML feature with a time at which the observation took place, and with a value for the observation.

A reference system provides a scale of measurement for assigning values "to a location, time or other descriptive quantity or quality".

A coordinate reference system consists of a set of coordinate system axes that is related to the earth through a datum that defines the size and shape of the earth. Geometries in GML indicate the coordinate reference system in which their measurements have been made. The "parent" geometry element of a geometric complex or geometric aggregate makes this indication for its constituent geometries.

A temporal reference system provides standard units for measuring time and describing temporal length or duration. Following ISO 8601, the Gregorian calendar with UTC is used in GML as the default temporal reference system.

A Units of Measure (UOM) dictionary provides definitions of numerical measures of physical quantities, such as length, temperature, and pressure, and of conversions between UOMs.

## **Submitting organizations and Contacts:**

John Bobbitt, POSC

David Burggraf, Galdos Systems

Simon Cox, CSIRO

Paul Daisey, U.S. Census Bureau Geography Division

John Herring, Oracle Corporation

Sandra Johnson, MapInfo Corporation

Ron Lake, Galdos Systems

Richard Martell, Galdos Systems

Akifumi Nakai, NTT Data

Clemens Portele, Interactive Instruments

Milan Trninic, Galdos Systems

Paul Watson, Laser-Scan Ltd

Arliss Whiteside, BAE Systems Mission Solutions

# OpenGIS® Geography Markup Language (GML) Implementation Specification – Version 3.1.0

## 1 Scope

The Geography Markup Language (GML) is an XML encoding in compliance with ISO 19118 for the transport and storage of geographic information modelled according to the conceptual modelling framework used in the ISO 19100 series and including both the spatial and non-spatial properties of geographic features.

This specification defines the XML Schema syntax, mechanisms, and conventions that:

- Provide an open, vendor-neutral framework for the definition of geospatial application schemas and objects;
- Allow profiles that support proper subsets of GML framework descriptive capabilities;
- Support the description of geospatial application schemas for specialized domains and information communities;
- Enable the creation and maintenance of linked geographic application schemas and datasets;
- Support the storage and transport of application schemas and data sets;
- Increase the ability of organizations to share geographic application schemas and the information they describe.

Implementers may decide to store geographic application schemas and information in GML, or they may decide to convert from some other storage format on demand and use GML only for schema and data transport.

**NOTE** The model is conformant with the parts of the Abstract Specification of OGC and the ISO 19100 series which are listed in the normative references. For other parts of this specifications no abstract specification was available.

## 2 Conformance

Conformance with this specification shall be checked using all the relevant tests specified in Annex A (normative). The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance are specified in ISO 19105: Geographic information — Conformance and Testing.

In order to conform to this OpenGIS® Implementation Specification, a software implementation shall choose to implement:

- a) Any one of the conformance levels specified in Annex B (normative).

### 3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 8601:2000, Data elements and interchange formats – Information interchange Representation of dates and times

ISO/TS 19103:—<sup>1</sup>, *Geographic Information – Conceptual Schema Language*

ISO 19105:2000, *Geographic information – Conformance and testing*

ISO 19107:2003, *Geographic Information – Spatial Schema*

ISO 19108:2002, *Geographic Information – Temporal Schema*

ISO 19109:—<sup>1</sup>, *Geographic Information – Rules for Application Schemas*

ISO 19115:2003, *Geographic Information – Metadata*

ISO 19117:—<sup>1</sup>, *Geographic Information – Portrayal*

ISO 19118:—<sup>1</sup>, *Geographic Information – Encoding*

ISO 19123:—<sup>1</sup>, *Geographic Information – Coverages*

ISO/TS 19139:—<sup>1</sup>, *Geographic Information – Metadata – Implementation Specification*

OpenGIS® Abstract Specification Topic 0, *Overview*, OGC document 99-100r1

OpenGIS® Abstract Specification Topic 1, *Feature Geometry*, OGC document 01-101

OpenGIS® Abstract Specification Topic 2, *Spatial referencing by coordinates*, OGC document 03-071r1

OpenGIS® Abstract Specification Topic 5, *The OpenGIS Feature*, OGC document 99-105r2

OpenGIS® Abstract Specification Topic 8, *Relations between Features*, OGC document 99-108r2

OpenGIS® Abstract Specification Topic 10, *Feature Collections*, OGC document 99-110

IETF RFC 2396, *Uniform Resource Identifiers (URI): Generic Syntax*. (August 1998)

IETF RFC 2732, *Format for Literal IPv6 Addresses in URLs*. (December 1999)

W3C XLink, *XML Linking Language (XLink) Version 1.0*. W3C Recommendation (27 June 2001)

W3C XMLName, *Namespaces in XML*. W3C Recommendation (14 January 1999)

W3C XMLSchema-1, *XML Schema Part 1: Structures*. W3C Recommendation (2 May 2001)

W3C XMLSchema-2, *XML Schema Part 2: Datatypes*. W3C Recommendation (2 May 2001)

W3C Xpointer, *XML Pointer Language (XPointer) Version 1.0*. W3C Working Draft (16 August 2002)

---

<sup>1</sup> To be published.

W3C XML Base, *XML Base*, W3C Recommendation (27 June 2001)

W3C XML, *Extensible Markup Language (XML) 1.0 (Second Edition)*, W3C Recommendation 6 October 2000

W3C SVG, *Scalable Vector Graphics (SVG) 1.0 Specification*. W3C Recommendation (04 September 2001)

W3C SMIL, *Synchronized Multimedia Integration Language (SMIL 2.0)*. W3C Recommendation (07 August 2001)

*The Schematron Assertion Language 1.5.* Rick Jelliffe 2002-10-01

NOTE Schematron is currently undergoing ISO standardization to become ISO/IEC 19757 - DSDL Document Schema Definition Language – Part 3: Rule-based validation – Schematron.

NOTE Due to differences with the current version of ISO 19111 (Geographic Information – Spatial referencing by coordinates) this standard is currently not in the list of normative references. See annex D.

## 4 Terms and definitions

### 4.1

#### **application schema**

conceptual schema for data required by one or more applications [ISO 19101]

### 4.2

#### **GML application schema**

an XML Schema written according to the GML rules for application schemas and which defines a vocabulary of geographic objects for a particular domain of discourse

### 4.3

#### **association**

semantic relationship between two or more classifiers that specifies connections among their instances [ISO 19103]

### 4.4

#### **attribute <XML>**

an information item in the XML Information Set [InfoSet]

NOTE 1 In this document an attribute is an XML attribute unless otherwise specified. The syntax of an XML attribute is “Attribute ::= Name = AttValue”. An attribute typically acts as an XML element modifier (e.g. <Road gml:id = “r1” /> here gml:id is an attribute).

NOTE 2 ISO 19130 defines the term “attribute” as: property which describes a geometrical, topological, thematic, or other characteristic of an entity. This definition is wider than the term “attribute” as it is used in this International Standard and corresponds more to a “GML property”.

### 4.5

#### **boundary**

set that represents the limit of an entity [ISO 19107]

### 4.6

#### **category**

One of a set of classes in a classification scheme.

### 4.7

#### **child <XML>**

an XML element c that is in the content of another element p, its parent, but is not in the content of any other element in the content of p

**4.8****closure**

union of the **interior** and **boundary** of a **topological** or **geometric object** [ISO 19107]

**4.9****codelist**

Value domain including a code for each permissible value

**4.10****codespace**

Rule or authority for a code, name, term or category

EXAMPLE Examples of a codespace include dictionaries, authorities, codelists, etc.

**4.11****composite curve**

**sequence** of **curves** such that each curve (except the first) starts at the end point of the previous curve in the **sequence** [ISO 19107]

NOTE A composite curve, as a set of direct positions, has all the properties of a curve.

**4.12****composite solid**

connected **set** of **solids** adjoining one another along shared **boundary surfaces** [ISO 19107]

NOTE A composite solid, as a set of direct positions, has all the properties of a solid.

**4.13****composite surface**

connected **set** of **surfaces** adjoining one another along shared **boundary curves** [ISO 19107]

NOTE A composite surface, as a set of direct positions, has all the properties of a surface.

**4.14****control point**

one of a sequence of points defining the position of a curve segment or surface patch – together with additional information about the interpolation method

NOTE The controlPoints of a curve segment are used to control its shape, but are not always on the curve segment itself.

**4.15****coordinate**

one of a **sequence** of numbers designating the position of a **point** in n-dimensional space [ISO 19111]

NOTE In a coordinate reference system, the n numbers shall be qualified by units.

**4.16****coordinate reference system**

**coordinate system** that is related to the real world by a datum [ISO 19111]

**4.17****coordinate system**

set of mathematical rules for specifying how **coordinates** are to be assigned to **points** [ISO 19111]

**4.18****coordinate tuple**

**tuple** composed of coordinates

#### **4.19**

##### **coverage**

feature that acts as a function to return values from its range for any direct position within its spatiotemporal domain [ISO 19123]

#### **4.20**

##### **curve**

1-dimensional **geometric primitive**, representing the continuous image of a line

**NOTE** The boundary of a curve is the set of points at either end of the curve. If the curve is a cycle, the two ends are identical, and the curve (if topologically closed) is considered to not have a boundary. The first point is called the start point, and the last is the end point. Connectivity of the curve is guaranteed by the "continuous image of a line" clause. A topological theorem states that a continuous image of a connected set is connected.

#### **4.21**

##### **data type**

specification of a value domain with operations allowed on values in this domain [ISO 19103]

**EXAMPLE** Integer, Real, Boolean, String, Date (conversion of a data into a series of codes).

**NOTE** Data types include primitive predefined types and user-definable types.

#### **4.22**

##### **datum**

parameter or set of parameters that serve as a reference or basis for the calculation of other parameters [ISO 19111]

**NOTE 1** A datum defines the position of the origin, the scale, and the orientation of the axes of a coordinate system.

**NOTE 2** A datum may be a geodetic datum, a vertical datum or an engineering datum.

#### **4.23**

##### **direct position**

position described by a single set of coordinates within a coordinate reference system [ISO 19107]

#### **4.24**

##### **domain**

well-defined **set** [ISO 19103]

**NOTE 1** A mathematical **function** can be defined on this set, i.e. in a function  $f:A \rightarrow B$  A is the domain of the function f.

**NOTE 2** A domain as in domain of discourse refers to a subject or area of interest.

#### **4.25**

##### **edge**

1-dimensional topological primitive [ISO 19107]

#### **4.26**

##### **element <XML>**

an information item in the XML information Set

**NOTE** From the XML Information Set Specification: Each XML document contains one or more elements, the boundaries of which are either delimited by start-tags and end-tags, or, for empty elements, by an empty-element tag. Each element has a type, identified by name, sometimes called its "generic identifier" (GI), and may have a set of attribute specifications. Each attribute specification has a name and a value.

#### **4.27**

##### **exterior**

difference between the universe and the **closure** [ISO 19107]

**4.28****face**

2-dimensional **topological primitive** [ISO 19107]

NOTE The geometric realization of a face is a surface. The boundary of a face is the set of directed edges within the same topological complex that are associated to the face via the boundary relations. These can be organized as rings

**4.29****feature**

abstraction of real world phenomena [ISO 19101]

NOTE A feature may occur as a type or an instance. Feature type or feature instance should be used when only one is meant

**4.30****feature relationship**

association between **features** [ISO 19103]

**4.31****function**

rule that associates each element from a domain (source, or domain of the function) to a unique element in another domain (target, co-domain, or range) [ISO 19107]

**geodetic datum**

datum describing the relationship of a coordinate system to the Earth [ISO 19111]

NOTE In most cases, the geodetic datum includes an ellipsoid definition.

**4.33****geometric object**

spatial object representing a geometric set [ISO 19107]

**4.34****geometric set**

set of direct positions [ISO 19107]

**4.35****geometry property <GML>**

Property of a GML feature that describes some aspect of the geometry of the feature.

NOTE The geometry property name is the role of the geometry in relation to the feature.

**4.36****grid**

network composed of two or more sets of **curves** in which the members of each set intersect the members of the other sets in an algorithmic way [ISO 19123]

NOTE The curves partition a space into grid cells.

**4.37****interior**

set of all **direct positions** that are on a **geometric object** but which are not on its **boundary** [ISO 19107]

**4.38****line string**

curve composed of straight-line segments

**4.39****measure <GML>**

A value described using a numeric amount with a scale or using a scalar reference system

NOTE When used as a noun, measure is a synonym for quantity.

#### 4.40

##### **measurand**

Phenomenon or property that is subject to observation.

#### 4.41

##### **namespace <XML>**

Collection of names, identified by an URI reference which are used in XML documents as element names and attribute names [XML]

#### 4.42

##### **node**

0-d topological primitive [ISO 19107]

#### 4.43

##### **object**

entity with a well defined boundary and identity that encapsulates state and behaviour [ISO 19107]

NOTE A GML object is an XML element of a type derived from AbstractGMLType.

#### 4.44

##### **observable**

Phenomenon or property that is subject to observation

#### 4.45

##### **point**

0-dimensional **geometric primitive**, representing a position [ISO 19107]

NOTE The boundary of a point is the empty set.

#### 4.46

##### **polygon**

a planar **surface** defined by 1 exterior boundary and 0 or more interior boundaries

#### 4.47

##### **property <GML>**

a child element of a GML object.

NOTE It corresponds to feature attribute and feature association in ISO 19109. If a GML **property of a feature** has an xlink:href attribute that references a feature, the **property** represents a feature association.

#### 4.48

##### **quantity**

property ascribed to phenomena, bodies or substances that can be used to specify the size, amount, or extent of a particular phenomenon, body or substance [ISO 19126]

NOTE In GML a quantity is always a value described using a numeric amount with a scale or using a scalar reference system. Quantity is a synonym for measure when the latter is used as a noun.

#### 4.49

##### **range**

the set B of a mathematical function ( $f:A \rightarrow B$ ) is called the range of the function.

#### 4.50

##### **rectified grid**

**grid** for which there is an affine transformation between the grid coordinates and the **coordinates** of an external **coordinate reference system** [ISO 19123]

**4.51****schema**

formal description of a model [ISO 19101]

NOTE In general, a schema is an abstract representation of an **object's** characteristics and relationship to other **objects**. An XML schema represents the relationship between the attributes and elements of an XML object (for example, a document or a portion of a document)

**4.52****semantic type**

A category of objects that share some common characteristics and are thus given an identifying type name in a particular domain of discourse.

**4.53****sequence**

finite, ordered collection of related items (objects or values) that may be repeated [ISO 19107]

**4.54****set**

unordered collection of related items (**objects** or values) with no repetition [ISO 19107]

**4.55****spatial object**

**object** used for representing a spatial characteristic of a feature [ISO 19107]

**4.56****tag <XML>**

the text in an XML document bounded by angle brackets

EXAMPLE

<Road>.

NOTE A tag with no forward slash (e.g. <Road>) is called a start tag (also opening tag), and one with a forward slash (e.g. </Road>) is called an end tag (also closing tag).

**4.57****topological object**

**spatial object** representing spatial characteristics that are invariant under continuous transformations [ISO 19107]

**4.58****topology**

A branch of geometry describing the properties of a figure that are unaffected by continuous distortion [Collins Concise Dictionary]

NOTE

Topology is mostly concerned with identifying the connectivity of networks and the adjacency of surfaces.

**4.59****tuple**

An ordered list of values

**4.60****type**

A class in a classification system

NOTE

See also data type.

**4.61****Uniform Resource Identifier (URI)**

A unique identifier, usually taking the form of a short string or address that is used to identify the location of a resource

NOTE The general syntax is <schema>::<scheme-specific-part>. The hierarchical syntax with a namespace is <schema>://<authority><path>?<query> - see [RFC 2396].

## 5 Conventions

### 5.1 Deprecated parts of previous versions of GML

The verb "**deprecate**" provides notice that the referenced portion of the specification is being retained for backwards compatibility with earlier versions but may be removed from a future version of the specification without further notice.

### 5.2 Symbols (and abbreviated terms)

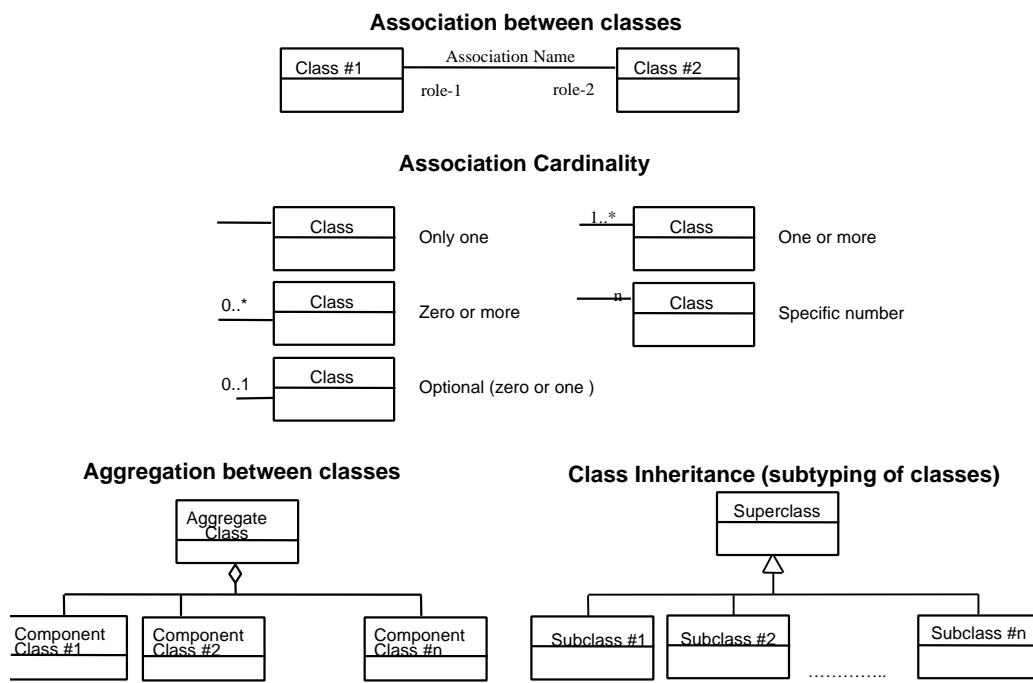
The following symbols and abbreviated terms are used in this document:

CRS	Coordinate Reference System
CSV	Comma Separated Values
CT	Coordinate Transformation
DTD	Document Type Definition
EPSG	European Petroleum Survey Group
GIS	Geographic Information System
GML	Geography Markup Language
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
iff	if and only if
ISO	International Organization for Standardization
OGC	Open GIS Consortium
RDF	Resource Description Framework
RFC	Request for Comments
SMIL	Synchronized Multimedia Integration Language
SOAP	Simple Object Access Protocol
SVG	Scalable Vector Graphics
UML	Unified Modeling Language
URL	Uniform Resource Locator
WKT	Well-Known Text

WFS	Web Feature Service
XML	Extensible Markup Language
XSLT	eXtensible Stylesheet Language – Transformations
0D	Zero Dimensional
1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional

### 5.3 UML Notation

Many diagrams that appear in this standard are presented using the Unified Modeling Language (UML) static structure diagram. The UML notations used in this standard are described in the diagram below.



**Figure 1 — UML notation**

In this standard, the following stereotypes of UML classes are used:

- <<DataType>> A descriptor of a set of values that lack identity (independent existence and the possibility of side effects). A DataType is a class with no operations whose primary purpose is to hold the information.
- <<CodeList>> is a flexible enumeration that uses string values for expressing a list of potential values.
- <<Enumeration>> is a fixed list of valid identifiers of named literal values. Attributes of an enumerated type can only take values from this list.

- d) <>Union>> is a list of attributes. The semantics is that only one of the attributes can be present at any time.

In this standard, the following standard data types are used:

- a) CharacterString – A sequence of characters (in general this data type is mapped to “string” in XML Schema)
- b) Integer – An integer number (in general this data type is mapped to “integer” in XML Schema)
- c) Real – A floating point number (in general this data type is mapped to “double” in XML Schema)
- d) Boolean – A value specifying TRUE or FALSE (in general this data type is mapped to “boolean” in XML Schema)

## 5.4 XML Schema

The normative parts of the specification use the W3C XML Schema language to describe the grammar of conformant GML data instances. XML Schema is a rich language with many capabilities and subtleties. While a reader who is unfamiliar with XML Schema may be able to follow the description in a general fashion, this specification is not intended to serve as an introduction to XML Schema. In order to have a full understanding of this specification it is necessary for the reader to have a reasonable knowledge of XML Schema.

# 6 Overview of GML Schemas

## 6.1 Review of GML version 2 – XML for Simple Features

The previous major release of GML (version 2.x) was concerned with what the OGC calls simple features: features whose geometric properties are restricted to 'simple' geometries for which coordinates are defined in two dimensions and the delineation of a curve is subject to linear interpolation. These simple features in GML could be used to represent real-world phenomena. The task to which a digital representation will ultimately be put guides the classification of real world phenomena, which in turn determines the feature types that need to be defined.

For example, a city could be represented as a feature collection where the individual features represent such things as rivers, roads and colleges. Each of these feature types would have named, typed properties. The River feature type might have a property called name whose value shall be of the type 'string'. It is common practice to refer to the typed property; thus the River feature type is said to have a string property called name. Similarly, the Road feature type might have a string property called classification and an integer property called number. Properties with simple types (e.g. integer, string, float, boolean) are collectively referred to as simple properties.

The features required to represent a city might have geometry-valued properties as well as simple properties. Just like other properties, geometric properties shall be named. So the River feature type might have a geometric property called centerLineOf and the Road feature type might have a geometric property called linearGeometry. Just as it is common to have multiple simple properties defined on a single feature type, so too a feature type may have multiple geometric properties; the College feature type might have both a point property called location and a polygon property called campus.

## 6.2 GML version 3 – more than Simple Features, plus ISO conformance

This version of GML addresses the following needs that were not addressed or adequately met by version 2:

- represent geospatial phenomena in addition to simple 2D linear features, including features with complex, non-linear, 3D geometry, features with 2D topology, features with temporal properties, dynamic features, coverages, and observations;
- provide more explicit support for properties of features and other objects whose value is complex;
- represent spatial and temporal reference systems, units of measure and standards information;
- use reference system, units and standards information in the representation of geospatial phenomena, observations, and values;
- represent default styles for feature and coverage visualization;
- conform with standards from the ISO 19100 series.

The expansion of GML to meet these needs is reflected in base schemas for GML version 3 that are over eight times as large as the base schemas for GML version 2. However, few applications will use all of the definitions added to GML version 3. Implementers may use a selective subset of the GML version 3 schemas sufficient to support the definitions in their application schemas. Methods for modular use of GML are discussed in clause 20; a schema subsetting tool is provided in Annex G.

### 6.3 Backward compatibility

GML version 3.1.0 maintains backward compatibility for GML version 3.0.0 and 2.1.2 **instance documents** by preserving, but deprecating, some schema components that have been replaced by different constructs in the current version. GML version 2.1.2 and version 3.0.0 application schemas will in most cases require only minor changes to upgrade to GML version 3.1.0, removing the use of such deprecated schema components.

Deprecated GML schema components may be eliminated without notice from a future GML version.

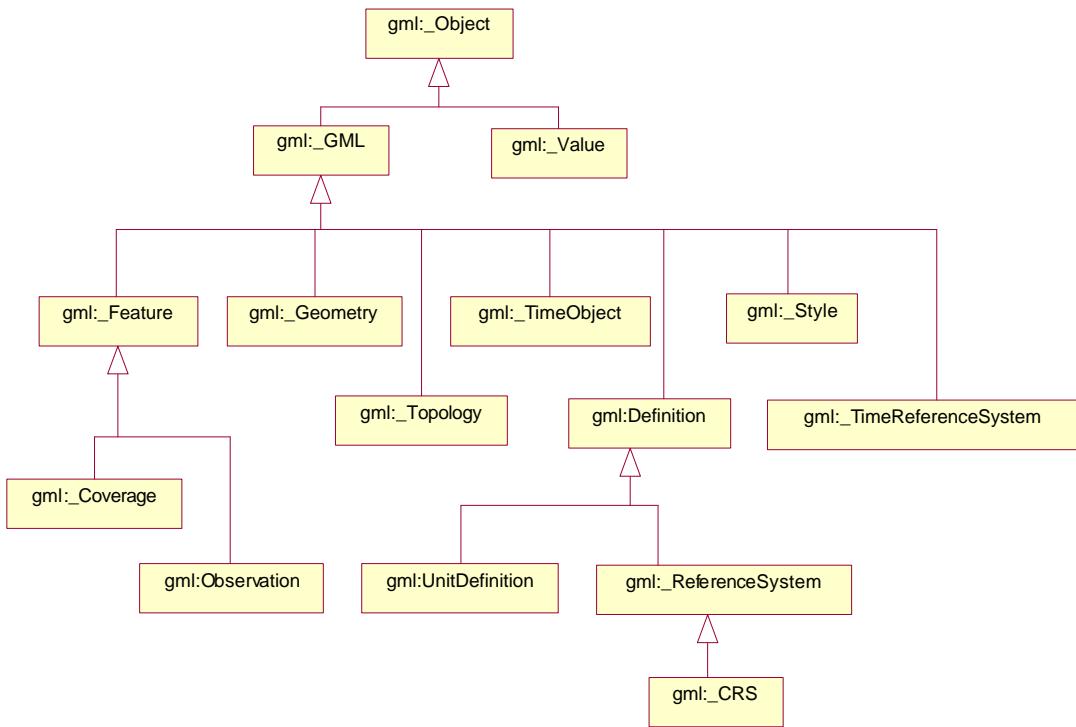
Deprecated GML schema components are written in *italics*.

### 6.4 GML 3 Application Schemas

Designers of GML Application schemas may extend or restrict the types defined in the GML base schemas to define appropriate types for an application domain. They may also use non-abstract elements and attributes from the GML base schemas as-is in an application schema if no changes are required. GML Application schemas may be constructed by hand using a text editor or specialized XML/Schema editor, in effect using XML/Schema as a conceptual schema language. GML Application schemas may also be constructed as part of a model driven process by automated translation to XML/Schema from conceptual models defined in other conceptual schema languages such as UML. In both cases, conformance with the requirements of this specification requires application schema use of all of the applicable GML base schema constructs, directly or by specialization, application schema validity according to the rules for XML/Schema, and application instance document data conformance to the application schema definitions verification by a validating XML parser. How the GML application schemas were produced is irrelevant for conformance to the requirements of this specification.

### 6.5 How GML 3 components are factored into schema documents

GML defines the various entities such as features, geometries, topologies etc. through a hierarchy of GML objects as shown in the UML diagram in Figure 2.



**Figure 2 — GML Class Hierarchy**

The items in Figure 2 with a preceding underscore (e.g. Feature) can be viewed as representative object instances of the class. The element `gml:_Feature`, for example, is to be interpreted as “any GML feature”.

The normative schemas of GML are organized in this specification around these GML object types.

Clause 7.2 describes the Xlink schema. This schema is an OGC implementation of the XLink specification using XML Schema. It may be replaced in some future release by an equivalent schema from the W3C.

Clause 7.3 defines the GML representation of some basic data types that are used in different GML Schemas. Most of these types are simple types or simple content types.

Clause 7.5 describes the schema `gmlBase.xsd`. This defines the root object (`gml:_Object`), the root of the GML class hierarchy (`gml:_GML`) and the root for MetaData property packages (`gml:_MetaData`).

Clause 8 describes the feature schema which defines `gml:_Feature` and some derived components.

Clauses 9, 10 and 11 describe the geometry schemas that define `gml:_Geometry` and some derived components.

Clause 12 describes the Coordinate Reference System schemas that define the sub-types of `gml:_ReferenceSystem`, and the elements and types required to construct specific Coordinate Reference Systems.

Clause 13 describes the schemas for Topology which define `gml:_Topology` and some derived components.

Clause 14 describes the schemas for temporal constructs and for dynamic features.

Clause 15 describes the schemas for definitions and dictionaries

Clause 16 describes the schemas for the construction of units of measure, measures and value objects.

Clause 17 describes the schemas for the description of Direction.

Clause 18 describes the schemas for Observations.

Clause 19.2 describes the schemas for grid geometries.

Clause 19.3 describes the schemas for coverages. This describes `gml:_Coverage` and some derived components.

Clause 20 describes the schemas for the graphical default styling of GML features, geometries and topologies.

## 6.6 GML schemas to import into your GML 3 Application Schema

Most applications will make use of only a subset of the schemas described in this specification. Schema dependencies are discussed in greater detail in Clause 21. As a starting point, consider the following;

- a) If you are modelling geographic features you will need the `feature.xsd`.
- b) If your features have properties which make use of units of measure you will need to import one of the `basicTypes.xsd`, `measure.xsd` and `valueObjects.xsd`, but you will not need the `units.xsd` schema unless you are defining units of measure such as appear in a units of measure dictionary.
- c) If you need only simple 1D or 2D geometries you need only the `geometryBasic0d1d` and `geometryBasic2d` schemas. You need the additional geometry schemas only if you require support for complex, 3D or non-linear geometry types. This is discussed in greater detail in Clauses 9, 10 and 11.
- d) You only need `topology.xsd` if your features have topology properties.
- e) You need the Coordinate Reference System schemas only if you are constructing Coordinate Reference System dictionary entries and those for supporting components (e.g. Prime Meridians, Geodetic Datums etc.).
- f) You need the temporal schemas only if you are concerned with time dependent feature properties or dynamic features.
- g) You need the coverage schemas only if you are constructing coverages (e.g. remotely sensed images, aerial photographs, soil distribution, digital elevation models).
- h) You need the observation schema only if you are concerned with modelling acts of observation such as taking photographs or making measurements. In the latter case you will also likely use the `valueObjects.xsd` and `measure.xsd` schemas.
- i) You need direction schema only to describe direction constructs such as compass bearings. The direction schema is included by the `observation.xsd` schema in order to support directed observations.
- j) You need the `defaultStyle.xsd` only if you are concerned with the description of graphical styles for features, geometries and topologies.
- k) The `Metadata` element in `gmlBase.xsd` is used to define packages of metadata properties that can be attached to any resource including GML features, geometries, and topologies.

In many applications you will only need to import the `feature.xsd` as this transitively imports the simple geometry schemas and `gmlBase.xsd`. For a thorough discussion of schema dependencies and modularity see Clause 21.

## 7 GML schemas – general rules and base schemas

### 7.1 Introduction, namespaces, versioning

This clause describes the normative GML schema documents and explains their contents, structure and dependencies.

The components described in the `xlinks.xsd` schema document are in the `http://www.w3.org/1999/xlink` namespace, for which the prefix `xlink` is normally used.

All the other schema documents describe components in the `http://www.opengis.net/gml` namespace, for which the prefix `gml` is normally used.

Each schema document in GML carries a `version` attribute as defined in the XML Schema Recommendation. The format of the `version` attribute string is `x.y.z` where `x` denotes the major version number, `y` denotes a minor version number and `z` denotes a bug fix release for that document. The current version is `3.1.0`.

### 7.2 Xlinks – Object Associations and Remote Properties

The normative `xlink` specification is available from W3C [xlink]. A schema document `xlinks.xsd` is provided as part of GML, pending the availability of a normative W3C XML Schema implementation.

Xlink components are used in GML to implement associations between objects by reference. GML property elements (see clause 7.5.3) may carry `xlink` attributes, which support the encoding of an association relationship by reference, the name of the property element denoting the target role in the association. The most important xlink component is:

`href`: Identifier of the resource which is the target of the association, given as a URI

The appearance of an `xlink:href` on a GML property indicates that the value of the property is to be found by traversing the link, that is the value is pointed to by the value of the `xlink:href` attribute. Following the terminology of Xlink, GML properties with `xlink:href` attributes are sometimes referred to as remote properties.

The other xlink components are used to indicate additional semantics of the relationship. The most useful of these are

`role`: description of the nature of the target resource, given as a URI

`arcrole`: description of the role or purpose of the target resource in relation to the present resource, given as a URI

`title`: description of the association or the target resource, given as text.

For complete definitions of these and other xlink components, including their use in extended xlink association maps, refer to the xlink specification [xlink].

In the GML core schemas (those defined in Clauses 7 to 20 of this international standard), simple xlink are used exclusively to denote object, feature or geometry associations and to denote remotely referenced property values.

### 7.3 Basic Types

#### 7.3.1 Overview

W3C XML Schema provides a set of built-in “simple” types which define methods for representing values as literals without internal markup. These are described in Part 2 of the W3C XML Schema recommendation. Because GML is an XML encoding in which instances are described using XML Schema, these simple types

are used as far as possible and practical for the representation of data types. W3C XML Schema also provides methods for defining (i) new simple types by restriction and combination of the built-in types, and (ii) complex types, with simple content, but which also have XML attributes. In many places where a suitable built-in simple type is not available, simple content types derived using the XML Schema mechanisms are used for the representation of data types in GML.

A set of these simple content types that are required by several GML components are defined in the basicTypes schema document, as well as some elements based on them,. These are primarily based around components needed to record amounts, counts, flags and terms, together with support for exceptions or null values. The basic types and elements described in the basicTypes schema listed in Annex C. The schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:basicTypes:v3.1.0

### 7.3.2 Convenience types

#### 7.3.2.1 gml:NullType

gml:NullType defines a content model that allows recording of a typed exception.

```
<simpleType name="NullEnumeration">
  <restriction base="string">
    <enumeration value="inapplicable"/>
    <enumeration value="missing"/>
    <enumeration value="template"/>
    <enumeration value="unknown"/>
    <enumeration value="withheld"/>
  </restriction>
</simpleType>

<simpleType name="NullType">
  <union memberTypes="gml:NullEnumeration anyURI"/>
</simpleType>
```

NullType is a union of the following enumerated values:

**inapplicable** there is no value

**missing** the correct value is not readily available to the sender of this data. Furthermore, a correct value may not exist

**template** the value will be available later

**unknown** the correct value is not known to, and not computable by, the sender of this data. However, a correct value probably exists

**withheld** the value is not divulged

and

**anyURI** which should refer to a resource which describes the reason for the value not being available

A particular community may choose to assign more detailed semantics to the standard values provided. Alternatively, the URI method enables a specific or more complete explanation for the absence of a value to be provided elsewhere and indicated by-reference in an instance document.

NullType is used as a member of a union in a number of simpleContent types defined below (clauses 7.3.2.3, 7.3.3.3, 7.3.4.3) where it is necessary to permit a value from the NullType union as an alternative to the primary type.

### 7.3.2.2 gml:Null

A convenience element gml:Null is declared as follows:

```
<element name="Null" type="gml:NullType"/>
```

This element might appear in data instance documents as follows:

```
<gml:Null>withheld</gml:Null>  
<gml:Null>http://my.big.org/explanations/theDogAteIt</gml:Null>
```

The first example uses one of the built-in values for Null. The second example contains a reference to an explanation available elsewhere, identified by a URI.

The purpose in providing the gml:Null element is as follows. In order to construct a content model where a value may be omitted, the cardinality constraint expressed in XML Schema using the construction ‘minOccurs=”0”’ might be used. However, this approach carries the risk that the reason for the value not being present may be misinterpreted. As an alternative the element gml:Null may be included as a member of a <choice> group, alongside an element of the datatype of a “normal” value. For example, the content model described by the schema fragment

```
<element name="footprint">  
  <complexType>  
    <choice>  
      <element ref="gml:Envelope"/>  
      <element ref="gml:Null"/>  
    </choice>  
  </complexType>  
</element>
```

allows either of the following data instances to be valid:

```
<footprint>  
  <gml:Envelope> ... </gml:Envelope>  
</footprint>  
  
<footprint>  
  <gml:Null>inapplicable</gml:Null>  
</footprint>
```

This allows the hypothetical element “footprint” to appear in an instance document, optionally containing an explicit marker indicate why it has no value, instead of having semantics inferred from the absence of a value.

The <choice> group is a useful content model structure available in the W3C XML Schema language, which shall be synthesised using more complex notation in some other conceptual schema languages (e.g. using a <<union>> stereotype, or an OCL constraint, in UML). It is an efficient and useful structuring component provided by this implementation language, so is used widely in GML as illustrated here and elsewhere.

### 7.3.2.3 gml:SignType

This is a convenience type with values “+” (plus) and “-” (minus).

```
<simpleType name="SignType">  
  <restriction base="string">  
    <enumeration value="-"/>  
    <enumeration value="+"/>  
  </restriction>  
</simpleType>
```

Elements or attributes of this type are used in various places, e.g. to indicate the direction of topological objects with "+" for forwards, or "-" for backwards.

### 7.3.3 Simple content

#### 7.3.3.1 gml:booleanOrNull, gml:doubleOrNull, gml:integerOrNull, gml:NameOrNull, gml:stringOrNull

A set of simple types provide extensions to the XML Schema built-in types boolean, double, integer, Name and string, to allow a choice of either the built-in simple type or a null. They are constructed as follows:

```
<simpleType name="booleanOrNull">
  <union memberTypes="gml:NullEnumeration boolean anyURI"/>
</simpleType>

<simpleType name="doubleOrNull">
  <union memberTypes="gml:NullEnumeration double anyURI"/>
</simpleType>

<simpleType name="integerOrNull">
  <union memberTypes="gml:NullEnumeration integer anyURI"/>
</simpleType>

<simpleType name="NameOrNull">
  <union memberTypes="gml:NullEnumeration Name anyURI"/>
</simpleType>

<simpleType name="stringOrNull">
  <union memberTypes="gml:NullEnumeration string anyURI"/>
</simpleType>
```

These are defined as unions of the respective XML Schema built-in simple type and the GML Nulltype. An element which uses one of these types may have content which is either boolean/string/Name/double/integer or a value from the Nulltype.

#### 7.3.3.2 gml:CodeType

This is a generalized type to be used for a term, keyword or name.

```
<complexType name="CodeType">
  <simpleContent>
    <extension base="string">
      <attribute name="codeSpace" type="anyURI" use="optional"/>
    </extension>
  </simpleContent>
</complexType>
```

It adds a XML attribute codeSpace to a term, where the value of the codeSpace attribute (if present) should indicate a dictionary, thesaurus, classification scheme, authority, or pattern for the term. As an example, the gmlBase schema contains an element declaration using this type (clause 7.5.6.3):

```
<element name="name" type="gml:CodeType"/>
```

so a corresponding element might appear in an instance document as follows :

```
<gml:name codeSpace = "http://www.ukusa.gov/placenames">St Paul</gml:name>
```

In this example "St Paul" is asserted to be a meaningful name according to http://www.ukusa.gov/placenames. Note that in all cases the rules for the values, including such things as uniqueness constraints, are set by the authority responsible for the codeSpace.

### 7.3.3.3 gml:MeasureType

A MeasureType supports recording Type is an amount encoded as a double, together with a units of measure indicated by a uom attribute. The value of uom (Units Of Measure) attribute is a reference to a reference system for the amount, often a ratio or interval scale.

```
<complexType name="MeasureType">
  <simpleContent>
    <extension base="double">
      <attribute name="uom" type="anyURI" use="required"/>
    </extension>
  </simpleContent>
</complexType>
```

The value of uom (Units Of Measure) attribute is a reference to a reference system for the amount, often a ratio or interval scale.

As an example, an application schema may contain an element declaration using this type

```
<element name = "height" type = "gml:MeasureType"/>
```

Also corresponding n elements of this type might appear in an data instance document as follows :

```
<height uom="http://www.iso.org/iso/en/.../units/m">1.4224</height>
<height uom="http://www.equestrian.org/units/hands">14</height>
<height uom="http://www.iso.org/iso/en/.../units/m">1.4224</height>
```

where the resource identified by the value of the uom attribute defines the unit of measure. GML components for this are defined in clause 16.2.

### 7.3.4 Lists

#### 7.3.4.1 gml:booleanList, gml:doubleList, gml:integerList, gml:NameList, gml:NCNameList, gml:QNameList, gml:booleanOrNullList, gml:NameOrNullList, gml:doubleOrNullList, gml:integerOrNullList

A set of types for lists of simple values are constructed according to the following patterns as follows:

```
<simpleType name="booleanList">
  <list itemType="boolean"/>
</simpleType>

<simpleType name=" doubleList">
  <list itemType="double"/>
</simpleType>

<simpleType name="integerList">
  <list itemType="integer"/>
</simpleType>

<simpleType name="NameList">
  <list itemType="Name"/>
</simpleType>

<simpleType name="NCNameList">
  <list itemType="NCName"/>
</simpleType>

<simpleType name="QNameList">
  <list itemType="QName"/>
</simpleType>
```

```

<simpleType name="booleanOrNullList">
  <list itemType="gml:booleanOrNull"/>
</simpleType>

<simpleType name="NameOrNullList">
  <list itemType="gml:NameOrNull"/>
</simpleType>

<simpleType name="doubleOrNullList">
  <list itemType="gml:doubleOrNull"/>
</simpleType>

<simpleType name="integerOrNullList">
  <list itemType="gml:integerOrNull"/>
</simpleType>

```

These types are defined as a list of values of the respective XML Schema built-in simple types, or of the Union types listed above. The \*OrNullList types support null values interspersed within a list.

An element which uses one of these types will contain a whitespace-separated list of members of the relevant type (see <http://www.w3.org/TR/xmlschema-2/#atomic-vs-list> for more details of the XML list structure).

**NOTE** None of list types defined here use an XML Schema string as an item. The reason for this is that a string may include embedded spaces, linefeeds, etc (<http://www.w3.org/TR/xmlschema-2/#string>). Since whitespace acts as the item separator in a list instance, there would be ambiguity in identifying items that potentially contain whitespace. On the other hand, an instance of the XML Schema Name type may not contain whitespace (<http://www.w3.org/TR/2000/WD-xml-2e-20000814#NT-Name>), so this may be used safely in a list context. The corollary of this is that if a term may contain whitespace, then such a term may not occur in a list instance.

#### 7.3.4.2 gml:CodeListType, gml:CodeOrNullListType

These two types provide for lists of terms. The schema definitions are as follows:

```

<complexType name="CodeListType">
  <simpleContent>
    <extension base="gml:NameList">
      <attribute name="codeSpace" type="anyURI" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

<complexType name="CodeOrNullListType">
  <simpleContent>
    <extension base="gml:NameOrNullList">
      <attribute name="codeSpace" type="anyURI" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

```

The values in an instance element of CodeListType should all be valid according to the rules of the dictionary, classification scheme, or authority identified by the value of its codeSpace attribute (e.g. list of localities, soil types, rock types, animal species etc.). As an example, an application schema may contain an element declaration using this type

```
<element name = "species" type = "gml:CodeListType"/>
```

so a corresponding element might appear in an instance document as follows :

```
<species codeSpace="http://my.big.org/florelegium">dryandra banksia hardenbergia lavender</species>
```

where the listed items are from “<http://my.big.org/florelegium>” which is a (hypothetical) list of flowers.

An instance element of CodeOrNullListType may also include embedded values from NullType. It is intended to be used in situations where a term or classification is expected, but the value may be absent for some reason.

#### 7.3.4.3 gml:MeasureListType, gml:MeasureOrNullListType

These two types provide for lists of quantities. The schema definitions are as follows:

```
<complexType name="MeasureListType">
  <simpleContent>
    <extension base="gml:doubleList">
      <attribute name="uom" type="anyURI" use="required"/>
    </extension>
  </simpleContent>
</complexType>

<complexType name="MeasureOrNullListType">
  <simpleContent>
    <extension base="gml:doubleOrNullList">
      <attribute name="uom" type="anyURI" use="required"/>
    </extension>
  </simpleContent>
</complexType>
```

As an example, an application schema may contain element declarations using these types

```
<element name = "heights" type = "gml:MeasureListType"/>
<element name = "weights" type = "gml:MeasureOrNullListType"/>
```

so corresponding elements might appear in an instance document as follows:

```
<heights uom="http://www.iso.org/iso/en/.../units/m">1.76 1.85 1.56 1.98</heights>
<weights uom="http://www.iso.org/iso/en/.../units/kg">67.0 73.4 withheld 85.1</weights>
```

In both examples all of the values in the list are described using the same scale. In the second example a null value appears where a measure is normally expected, but the value may be absent for some reason.

## 7.4 GML model and syntax

### 7.4.1 GML instance documents

GML uses an explicit syntax to instantiate a GML application schema conformant with the General Feature Model defined in ISO 19109 in an XML document.

A Feature (ISO 19109) is encoded as an XML element with the name of the feature type. Other identifiable objects are encoded as XML elements with the name of the object type.

Each feature attribute (ISO 19109) and feature association role (ISO 19109) is a property of a feature. Feature properties are encoded in an XML element.

**NOTE** The term "attribute" in XML refers to a specific syntactic component in XML documents, so to avoid confusion when describing the XML encoding, GML follows RDF (W3C, 1999) terminology and uses the term property rather than attribute or association role. The General Feature Model (ISO 19109) also uses the term "property" as a generalization of attributes, association roles and operations.

Furthermore, the property semantics, which is indicated by the name of the element representing the property, and the property value, which is given by the content of the property element, are distinguished. A property element may contain its value as content encoded inline, or reference its value with a simple XLink. The value of a property may be simple, or it may be a feature or other complex object. When recorded inline, the value

of a simple property is recorded as a literal value with no embedded markup (text), while if the value is complex it appears as a subtree using XML markup (i.e. an XML element with possible sub-structure).

**NOTE** The GML model has a relatively straightforward representation using the UML profile used in the ISO 19100 series (defined in ISO 19103). This is described in detail in Annex D and Annex E, but can be summarised approximately and briefly as follows.

Features are represented

- in UML by objects, where the name of the feature type is used as the name of the object class
- in GML instances by XML elements, where the name of the feature type is used as the name of the element

Feature properties are represented

- in UML by association roles with feature type classes, and attributes of feature type classes, where the property semantics are given by the association role name or attribute name
- in GML instances by sub-elements (known as property elements) of feature elements, where the property semantics are given by the property element name

The property value has a type indicated

- in UML by the class of the association target, or by the datatype of the attribute
- in GML, in the case of properties with complex values, by the name of the object element contained within the property element

The result is a layered XML document, in which XML elements corresponding to features, objects or values occur interleaved with XML elements corresponding to the properties that relate them. The function of a feature, object or value in context can always be determined by inspecting the name of the property element which directly contains it, or which carries the reference to it.

#### 7.4.2 Lexical conventions

There are several lexical conventions used in GML to assist in human comprehension of GML instances and schemas:

- objects are instantiated as XML elements with a conceptually meaningful name in UpperCamelCase;
- properties are instantiated as XML elements whose name is in lowerCamelCase;
- abstract elements have an underscore prepended to their \_name;
- the names of XML Schema complex types are in UpperCamelCase ending in the word “Type”;
- abstract XML Schema types have the word “Abstract” prepended.

#### 7.4.3 XML schema definition of GML languages

GML schema documents are W3C XML Schemas that define types and declare

- XML elements to encode GML objects with identity,
- XML elements to encode GML properties of those objects, and
- XML attributes qualifying those properties.

A GML object is an XML element of a type derived directly or indirectly from AbstractGMLType. From this derivation, a GML object may have a `gml:id` attribute.

A GML property may not be derived from AbstractGMLType, may not have a `gml:id` attribute, or any other attribute of XML type ID.

An element is a GML property if and only if it is a child element of a GML object.

No GML object may appear as the immediate child of a GML object.

Consequently, no element may be both a GML object and a GML property.

NOTE In this version of GML, the use of additional XML attributes in a GML application schema is discouraged.

## 7.5 **gmlBase schema**

The types and elements used to establish the GML model and syntax are described in the schema listed in Annex C. The schema is identified by the following location-independent name (using URN syntax):

`urn:opengis:specification:gml:schema-xsd:gmlBase:v3.1.0`

### 7.5.1 Goals of base schema

The schema document **gmlBase.xsd** defines components that establish the GML Model and Syntax: in particular

- a root XML type from which XML types for all GML objects should be derived,
- a pattern and components for GML properties,
- patterns for collections and arrays, and components for generic collections and arrays,
- components for associating metadata with GML objects,
- components for constructing definitions and dictionaries.

### 7.5.2 Base objects

#### 7.5.2.1 **gml:\_Object**

An abstract convenience element `gml:_Object` is declared as follows:

```
<element name="_Object" abstract="true">
```

This element has no type defined, and is therefore implicitly an XML Schema “anyType”. It is used as the head of an XML Schema substitution group which unifies `complexContent` and certain `simpleContent` elements used for datatypes in GML, including the `gml:_GML` substitution group.

NOTE `gml:_Object` is defined primarily to act as a variable in certain aggregate patterns where it is necessary to allow either elements in the `gml:_GML` substitution group, or certain `complexContent` or `simpleContent` elements to be valid in an instance.

A GML dataset (also called a data instance or data document) is a GML object. This GML object can in turn be a collection of GML objects

#### 7.5.2.2 **gml:\_GML, gml:AbstractGMLType**

The most basic components for constructing identifiable GML objects are described in the schema as follows:

```

<element name="_GML" type="gml:AbstractGMLType" abstract="true" substitutionGroup="gml:_Object"/>

<complexType name="AbstractGMLType" abstract="true">
  <sequence>
    <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
    <element ref="gml:description" minOccurs="0"/>
    <element ref="gml:name" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute ref="gml:id" use="optional"/>
</complexType>

```

The abstract element gml:\_GML is “any GML object having identity”. It acts as the head of an XML Schema substitution group, which may include any element which is a GML feature, or other object, with identity. This is used as a variable in content models in GML core and application schemas. It is effectively an abstract superclass for all GML objects.

The pairing of gml:\_GML and gml:AbstractGMLType shows a basic pattern used in the GML schemas, whereby each GML object type is represented by a global element declaration, which has an associated XML Schema type definition. The name of an element representing a GML Object indicates the conceptual meaning of the object. Generic element names in GML include gml:\_Object, gml:\_GML, gml:\_Feature, gml:\_Value, gml:\_Coverage, gml:\_Topology and gml:\_CRS. These other generic elements representing objects are defined elsewhere in this specification.

The child XML elements and XML attributes of a GML object are properties of that object. Thus an object represented by an gml:\_GML element has four properties: metaDataProperty, description, name and id. These are described in clause 7.5.5.

### 7.5.3 GML properties

The term “property” is used to refer to a GML property, which is any characteristic of a GML object. An element in a GML document or data stream is a GML property if and only if it is a child element of a GML object element. The meaning of each property is indicated by the name of the element that instantiates it.

GML Objects may have an unlimited number of properties, in addition to those inherited from gml:AbstractGMLType. A property may be defined to have either simple or complex content. A property with simple content has an XML Schema simpleContent type, as illustrated by the case of the standard property elements gml:description and gml:name. A property with complex content has an XML Schema complexContent type, as in the case of the standard property element gml:member.

Property elements may use two modes:

- o by value: the value of the property is represented directly, as the content of the property element. This method is used by the standard property gml:name and may be used for gml:description (see clause 7.5.6).
- o by reference: the value of the property is available elsewhere, and is identified by the value of an **xlink:href** attribute on the property element. This alternative method may be used for the standard property gml:description (see clause 7.5.6).

#### 7.5.3.1 gml:\_association, gml:AssociationType

To support the encoding of properties that may have complex content, a basic pattern for property elements is provided in the schema as follows:

```

<element name="_association" type="gml:AssociationType" abstract="true"/>

<complexType name="AssociationType">
  <sequence>
    <element ref="gml:_Object" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>

```

```
</complexType>
```

NOTE The declaration of gml:\_association and its accompanying type definition is provided for convenience, to act as a template or pattern for the construction of property elements in application schemas. There is no requirement for specific properties to use XML Schema type derivation from gml:AssociationType to create properties in a conformant GML application schema. This contrasts with the requirement that the content model for all identifiable objects shall derive from gml:AbstractGMLType, and for all features from gml:AbstractFeatureType

### 7.5.3.2 gml:AssociationAttributeGroup

XLink components are the standard method to support hypertext referencing in XML. An XML Schema attribute group, gml:AssociationAttributeGroup, is provided to support the use of Xlinks as the method for indicating the value of a property by reference in a uniform manner in GML. This attribute group is defined in the schema as follows:

In xlink.xsd (see clause 7.2):

```
<attributeGroup name="simpleLink">
  <attribute name="type" type="string" fixed="simple" form="qualified"/>
  <attribute ref="xlink:href" use="optional"/>
  <attribute ref="xlink:role" use="optional"/>
  <attribute ref="xlink:arcrole" use="optional"/>
  <attribute ref="xlink:title" use="optional"/>
  <attribute ref="xlink:show" use="optional"/>
  <attribute ref="xlink:actuate" use="optional"/>
</attributeGroup>
```

in gmlBase.xsd (see clause 7.5):

```
<attributeGroup name="AssociationAttributeGroup">
  <attributeGroup ref="xlink:simpleLink"/>
  <attribute ref="gml:remoteSchema" use="optional"/>
</attributeGroup>
```

The value of a GML property that carries an xlink:href attribute is the resource returned by traversing the link.

In addition to the simpleLink components, the additional attribute **remoteSchema**, is provided to indicate a schema which constrains the description of the remote resource referenced by the xlink. Note that all components in the attribute group are optional.

### 7.5.4 By value or by reference?

The gml:\_Object element in the content model for properties is optional. In combination with the component cardinalities in AssociationAttributeGroup this means that an element of this type may have a content element (in the substitution group headed by gml:\_Object) or xlink attributes. GML property elements which follow this pattern can be used to attach values either by value or by reference.

NOTE When used in the “by value” form, a GML property element corresponds with an implementation of a UML composition association. When used in the “by reference” form, a GML property element corresponds with an implementation of a UML aggregation association. The standard form, which supports either style, may be compared with the UML untyped association. The name of the property element corresponds to the rolename on the UML association, or to a UML attribute name. The name of the value element (gml:Point in the example here) corresponds to the class of the target of the association.

EXAMPLE A utility property provided for GML Features (described below in Clause 8.2.3) is “position”. This may be used to indicate a spatial location by-value as follows:

```
<gml:position>
  <gml:Point gml:id="point96" srsName="urn:EPSG:geographicCRS:62836405">
    <gml:pos>-31.936 115.834</gml:pos>
```

```
</gml:Point>
</gml:position>
```

which uses the gml:Point object as defined in the GML geometry schemas (described in Clause 9.2). The same property element may be used to indicate a location by reference as follows:

```
<gml:position xlink:href="http://my.big.org/locations/point53"/>
```

where "http://my.big.org/location/point53" identifies a point supplied by the service indicated.

However, a property element following this pattern may have no content or attributes, or it may have both content and attributes, and still be XML Schema-valid. It is not possible to constrain the co-occurrence of content or attributes, so we cannot use W3C XML Schema to restrict a property to be exclusively by-value or by-reference.

#### 7.5.4.1 gml:\_strictAssociation

The constraint may be described precisely using an auxiliary constraint language Schematron [schematron]. The abstract, global elements **\_association** and **\_strictAssociation** both use **AssociationType**, but the following schema fragments shows how an element declaration may include a Schematron constraint to limit the property to act in either by-value or by-reference mode, but not both.

```
<appinfo>
  <sch:title>Schematron validation</sch:title>
  <sch:ns prefix="gml" uri="http://www.opengis.net/gml"/>
  <sch:ns prefix="xlink" uri="http://www.w3.org/1999/xlink"/>
  <sch:pattern name="Check either href or content not both">
    <sch:rule abstract="true" id="hrefOrContent">
      <sch:report test="@xlink:href and (*|text())">
        Property element may not carry both a reference to an object and contain an object.</sch:report>
      <sch:assert test="@xlink:href | (*|text())">
        Property element shall either carry a reference to an object or contain an object.</sch:assert>
    </sch:rule>
  </sch:pattern>
</appinfo>

<element name="_strictAssociation" type="gml:AssociationType" abstract="true">
  <annotation>
    <appinfo>
      <sch:pattern name="refAndContent co-occurrence prohibited">
        <sch:rule context="gml:_strictAssociation">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
    <documentation>shall carry a reference to an object or contain an object but not both</documentation>
  </annotation>
</element>
```

NOTE Some XML validators will process the Schematron constraints automatically. Otherwise, the Schematron code can be treated merely as a formal description of the required constraint. It is included here primarily as an illustration of how this might be used for specific purposes by application schema developers.

#### 7.5.4.2 gml:\_reference, gml:ReferenceType

Finally, in order to support the encoding of properties whose value is provided remotely by-reference (i.e. implementing the UML aggregation association), the following components are provided:

```
<element name="_reference" type="gml:ReferenceType" abstract="true"/>

<complexType name="ReferenceType">
  <sequence/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
```

```
</complexType>
```

The element `gml:_reference` is abstract, and thus may be used as the head of a substitution group of more specific elements providing a value by reference.

### 7.5.5 Convenience properties and property types

#### 7.5.5.1 gml:member

A concrete property element named “member” with complex content is declared in `gmlBase`:

```
<element name="member" type="gml:AssociationType"/>
```

This is primarily to support the construction of object collections. In an instance document this element may be used in a data instance as follows:

```
<gml:member xlink:href="http://my.org/thingie#i456" gml:remoteSchema="http://my.org/schemas/thingie.xsd"/>
```

where the value is given by reference, or

```
<gml:member>
  <my:Thingie gml:id="i456" ... />
</gml:member>
```

where the value is encoded inline. Since `gml:_Object` is an abstract element, in this real instance it is replaced by a concrete element in its substitution group. The following element definition in the “`http://my.org/schemas/thingie.xsd`” application schema where `my:ThingyType` is defined supports this replacement:

```
<element name="Thingie" type="my:ThingyType" substitutionGroup="gml:_Object"/>
```

#### 7.5.5.2 gml:stringOrRefType

`gml:StringOrRefType` is provided to contain extended text values. It is defined in the schema document as follows:

```
<complexType name="StringOrRefType">
  <simpleContent>
    <extension base="string">
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </extension>
  </simpleContent>
</complexType>
```

It is an implementation of the GML property model in a `simpleContent` type.

NOTE This is possible because an empty string is schema-valid.

This type is available wherever there is a need for a “text” type property. It is of string type, so the text can be included inline, but the value can also be referenced remotely via `xlinks` which are in the `AssociationAttributeGroup`. If the remote reference is present, then the value obtained by traversing the link should be used if possible, and the string content of the element can be used for an annotation.

### 7.5.6 Standard properties of GML Objects

XML Schema types for all concrete GML objects derive from `gml:AbstractGMLType`. This means that all GML objects inherit certain standard properties that are included in the content model of `gml:AbstractGMLType`.

### 7.5.6.1 gml:metaDataProperty

```
<element name="metaDataProperty" type="gml:MetaDataPropertyType"/>
```

This property contains or refers to a metadata package that contains metadata properties. More detail is provided below in clause 7.5.8.

### 7.5.6.2 gml:description

```
<element name="description" type="gml:StringOrRefType"/>
```

The value of this property is a text description of the object. **description** uses the GML **StringOrRefType** defined in clause 7.5.5.2, so may contain a simple text string content, or carry a reference to an external description.

### 7.5.6.3 gml:name

```
<element name="name" type="gml:CodeType"/>
```

This property provides a label for the object, commonly a descriptive name.

An object may have several names, typically assigned by different authorities. **gml:name** uses the GML **CodeType** content model. The authority for a name is indicated by the value of its (optional) **codeSpace** attribute. The name may or may not be unique, as determined by the rules of the organization responsible for the codeSpace. In common usage there will be one name per authority, so a processing application may select the name from its preferred codeSpace.

### 7.5.6.4 gml:StandardObjectProperties

The three preceding property elements are collected in a model group **gml:StandardObjectProperties**, defined as follows:

```
<group name="StandardObjectProperties">
<sequence>
  <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
  <element ref="gml:description" minOccurs="0"/>
  <element ref="gml:name" minOccurs="0" maxOccurs="unbounded">
    <annotation>
      <documentation>Multiple names may be provided. These will often be distinguished by being assigned by different authorities, as indicated by the value of the codeSpace attribute. In an instance document there will usually only be one name per authority. </documentation>
    </annotation>
  </element>
</sequence>
</group>
```

This is provided for convenience in the construction of application schema, particularly when it is desired to define types derived by restriction from **gml:AbstractGMLType** and **gml:AbstractFeatureType**. Derivation by restriction requires that all components that are used unchanged are copied down into the new type definition. As an alternative to including three lines for **gml:metaDataProperty**, **gml:description** and **gml:name**, a one line reference to **gml:StandardObjectProperties** may be used instead:

```
<group ref="gml:StandardObjectProperties"/>
```

### 7.5.6.5 gml:id

```
<attribute name="id" type="ID"/>
```

This property supports provision of a database handle for the object. Its use is optional but recommended. It is of XML type **ID**, so is constrained to be unique in the XML document within which it occurs. An external

identifier for the object in the form of a URI may be constructed using standard methods [URI]. This is done by concatenating the URI for the document, a fragment separator “#”, and the value of the attribute of XML type ID.

### 7.5.7 Bags and arrays

#### 7.5.7.1 gml:ArrayAssociationType

For a property that can have multiple values all with the same relationship to the containing object, the cardinality of the association is extended as follows:

```
<complexType name="ArrayAssociationType">
  <sequence>
    <element ref="gml:_Object" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>

<element name="members" type="gml:ArrayAssociationType"/>
```

This property element may appear in a data instance as follows:

```
<gml:members>
  <my:Thingie gml:id="i456" />
  <my:Thingie gml:id="i457" />
  <my:Thingie gml:id="i458" />
</gml:members>
```

#### 7.5.7.2 gml:Array, gml:ArrayType, gml:Bag, gml:BagType

Two concrete object elements are provided for generic arrays and bags. The schema constructions are as follows:

```
<complexType name="ArrayType">
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <sequence>
        <element ref="gml:members" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="Array" type="gml:ArrayType" substitutionGroup="gml:_GML"/>
```

intended to be used for a collection whose member objects are of homogeneous type and where their order is significant, and

```
<complexType name="BagType">
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <sequence>
        <element ref="gml:member" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:members" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="Bag" type="gml:BagType" substitutionGroup="gml:_GML"/>
```

for general collections with no implication about the type, order or uniqueness of the member objects.

## 7.5.8 Metadata

### 7.5.8.1 `gml:metaDataProperty`, `gml:metaDataPropertyType`, `gml:_MetaData`, `gml:AbstractMetaDataType`

The metaDataProperty follows the property pattern and is used to contain or refer to metadata for GML objects. It is defined in the schema as follows:

```
<complexType name="MetaDataPropertyType">
  <choice minOccurs="0">
    <element ref="gml:_MetaData" minOccurs="0"/>
    <any processContents="lax"/>
  </choice>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
  <attribute name="about" type="anyURI" use="optional"/>
</complexType>

<element name="metaDataProperty" type="gml:MetaDataPropertyType"/>
```

The optional “about” attribute carries a URI which points to an element or range of elements, or other resource to which this metadata refers.

The value of the metaDataProperty is an abstract element `gml:_MetaData` that acts as a placeholder for “any package of metadata properties”, defined as follows:

```
<complexType name="AbstractMetaDataType" abstract="true" mixed="true">
  <attribute ref="gml:id" use="optional"/>
</complexType>

<element name="_MetaData" type="gml:AbstractMetaDataType" abstract="true" substitutionGroup="gml:_Object"/>
```

It is used as follows:

1. The user defines or re-uses an existing metadata schema. The structure of this metadata schema is more or less arbitrary but should satisfy the following pattern:
  - The metadata schema shall declare a single root element. This element shall be substitutable for `gml:_MetaData` – i.e. it is derived from `AbstractMetaDataType`
  - The content model of this root element shall be an XML Schema ComplexType that derives by extension from `gml:AbstractMetaDataType`.
  - The content model of this root element shall consist of an XML model group of elements each of which is a metadata property (e.g. age, accuracy, creation date etc.) of the object to which the `gml:metaDataProperty` is attached.
  - If the user wishes to more strongly restrict the value of the `gml:metaDataProperty`, the application schema designer shall define a new property that is an XML Schema restriction of `gml:metaDataProperty`.
2. The metaDataProperty in the data instance points to or includes the values of the metadata properties defined by the schema in 1. above.
3. The metaData property is used in one of two ways:
  - The metadata schema shall declare a single root element. This element shall be substitutable for `gml:_MetaData`.
  - Attached to a GML object and without the “`gml:about`” attribute it provides metadata for the GML object to which it is attached.

- Attached to a GML object collection and with the `gml:about` attribute it provides metadata for the GML objects within the object collection referenced by the `gml:about` attribute, the value of which is an XPointer expression.

Alternatively, any other existing metadata syntax in XML can be used instead of an element substitutable for `gml:_MetaData` due to the

```
<any processContents="lax"/>
```

declaration.

If metadata following the conceptual model of ISO 19115:2003 shall be encoded in a GML document, the corresponding Implementation Specification specified in ISO WD 19139 shall be used to encode the metadata information.

#### **7.5.8.2 `gml:GenericMetaData`, `gml:GenericMetaDataType`**

*For convenience, a generic concrete `MetaData` element has been provided in GML 3.0.0. This element is deprecated with GML 3.1.0.*

```
<complexType name="GenericMetaDataType" mixed="true">
  <complexContent mixed="true">
    <extension base="gml:AbstractMetaDataType">
      <sequence>
        <any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="GenericMetaData" type="gml:GenericMetaDataType" substitutionGroup="gml:_MetaData"/>
```

*This element has been intended to act as a container for metadata encoded using XML but defined in external schemas, in cases where it is not possible to assign the concrete components to the GML `_MetaData` substitution group. With GML 3.1.0 such metadata elements can be direct children of a `gml:metadataProperty` element.*

## **8 GML schemas – feature model**

### **8.1 General Concepts**

A GML feature is a meaningful object in the selected domain of discourse such as a Road, River, Person, Vehicle or Administrative Area. This follows the general definition of a feature given in ISO 19109 and the OGC Abstract Specification Topic 5.

### **8.2 Feature schema**

The feature schema, `feature.xsd`, provides a framework for the creation of GML features and feature collections. These are described in the schema listed in Annex C. The feature schema includes a basic geometry schema (`geometryBasic2d.xsd`).

The feature schema is identified by the following location-independent name (using URN syntax):

```
urn:opengis:specification:gml:schema-xsd:feature:v3.1.0
```

## 8.2.1 Features

### 8.2.1.1 **gml:AbstractFeatureType**

The basic feature model is given by the **gml:AbstractFeatureType**, defined in the schema as follows:

```
<complexType name="AbstractFeatureType" abstract="true">
<complexContent>
<extension base="gml:AbstractGMLType">
<sequence>
<element ref="gml:boundedBy" minOccurs="0"/>
<element ref="gml:location" minOccurs="0">
<annotation>
<appinfo>deprecated</appinfo>
<documentation>deprecated in GML version 3.1</documentation>
</annotation>
</element>
</sequence>
<attribute name="fid" type="string">
<annotation>
<appinfo>deprecated</appinfo>
<documentation>deprecated in GML version 3.0</documentation>
</annotation>
</attribute>
</extension>
</complexContent>
</complexType>
```

The content model for **gml:AbstractFeatureType** adds two specific properties suitable for geographic features to the content model defined in **gml:AbstractGMLType**. The value of the **gml:boundedBy** property describes an envelope that encloses the entire feature instance, and is primarily useful for supporting rapid searching for features that occur in a particular location.

*The value of the **gml:location** property describes the extent, position or relative location of the feature.*

**NOTE**      *gml:location is deprecated as part of the standard content model of gml:AbstractFeatureType.*

*GML features also have a (optional) fid attribute. This is for backwards compatibility with GML 2 and is deprecated in GML 3.*

### 8.2.1.2 **gml:\_Feature**

The element **gml:\_Feature** is declared as follows:

```
<element name="_Feature" type="gml:AbstractFeatureType" abstract="true" substitutionGroup="gml:_GML"/>
```

This abstract element serves as the head of a substitution group which may contain any elements whose content model is derived from **gml:AbstractFeatureType**. This may be used as a variable in the construction of content models.

**gml:\_Feature** can be thought of as “anything that is a GML feature” and can be used to define variables or templates in which the value of a GML property is “any feature”. This occurs in particular in a GML Feature Collection (see 8.2.7) where the **<gml:featureMember>** and **<gml:featureMembers>** properties contain one or multiple copies of **gml:\_Feature** respectively.

## 8.2.2 Standard feature properties

### 8.2.2.1 **gml:boundedBy**, **gml:BoundingShapeType**, **gml:EnvelopeWithTimePeriod**, **gml:EnvelopeWithTimePeriodType**

This property describes the minimum bounding box or rectangle that encloses the entire feature. Its content model is as follows:

```
<element name="boundedBy" type="gml:BoundingShapeType"/>
<complexType name="BoundingShapeType">
  <sequence>
    <choice>
      <element ref="gml:Envelope"/>
      <element ref="gml:Null"/>
    </choice>
  </sequence>
</complexType>
```

The **gml:Envelope** element is defined in clause 9.1.5. A value of **gml:Null** may appear if an extent is not applicable or not available for some reason.

**NOTE** Since an envelope is defined simply by the positions of two diagonally opposing corners, the exact footprint of an envelope depends on the coordinate reference system used. If the feature being described has zero extent, then the two corners will coincide and the envelope has zero size. As for all properties, it is the responsibility of the data provider to ensure that the value is correct.

**NOTE** *To provide backward compatibility with GML version 2, an additional element **gml:Box** is provided, with an identical content model and substitutable for **gml:Envelope**. This component is deprecated and should not be used in new applications.*

For envelopes that include a temporal extent, **gml:EnvelopeWithTimePeriod** is provided, defined as follows:

```
<complexType name="EnvelopeWithTimePeriodType">
  <complexContent>
    <extension base="gml:EnvelopeType">
      <sequence>
        <element ref="gml:timePosition" minOccurs="2" maxOccurs="2"/>
      </sequence>
      <attribute name="frame" type="anyURI" use="optional" default="#ISO-8601"/>
    </extension>
  </complexContent>
</complexType>

<element name="EnvelopeWithTimePeriod" type="gml:EnvelopeWithTimePeriodType"
  substitutionGroup="gml:Envelope"/>
```

This adds two **gml:timePosition** properties which describe the extent of a time-envelope.

Since **gml:EnvelopeWithTimePeriod** is assigned to the substitution group headed by **gml:Envelope**, it may be used whenever **gml:Envelope** is valid.

**NOTE** In common with all geometry elements derived from **gml:AbstractGeometryType** (clause 9.1.2.1), the coordinate reference system used for the positions defining the **gml:Envelope** may be indicated using the optional XML attribute “**srsName**”. If the coordinate reference system being used includes a time axis, then **gml:Envelope** can be used directly to describe a spatio-temporal extent.

### 8.2.2.2 **gml:location**, **gml:Location.PropertyType**, **gml:LocationKeyword**, **gml:LocationString**

The **gml:location** element is a convenience property that describes the generalized location of the feature. It is defined as follows:

```
<element name="location" type="gml:LocationPropertyType"/>

<complexType name="LocationPropertyType">
  <sequence>
    <choice>
      <element ref="gml:_Geometry"/>
      <element ref="gml:LocationKeyWord"/>
      <element ref="gml:LocationString"/>
      <element ref="gml:Null"/>
    </choice>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

The value of a location can be a geometry, a location string, a location keyword, or a null.

**NOTE** The flexible content model of the location property has proved to be difficult to implement in practice, so the element gml:location is deprecated in this version of GML.

A location string is text which should describe the location. It is declared as follows:

```
<element name="LocationString" type="gml:StringOrRefType"/>
```

The location keyword is a code usually selected from a controlled list. It is declared as follows:

```
<element name="LocationKeyWord" type="gml:CodeType"/>
```

**EXAMPLE** The following examples illustrate the different ways that the location property may appear in a data instance.

Location given as a gml Geometry in a particular spatial reference system:

```
<gml:location>
  <gml:Point gml:id="point96" srsName="urn:EPSG:geographicCRS:62836405">
    <gml:pos>31.936 115.834</gml:pos>
  </gml:Point>
</gml:location>
```

Location given using a name from a controlled source:

```
<gml:location>
  <gml:LocationKeyWord
    codeSpace="http://www.icsm.gov.au/icsm/cgna/index.html"
    >Leederville</gml:LocationKeyWord>
</gml:location>
```

Location given using a text string:

```
<gml:location>
  <gml:LocationString>Nigel Foster's town of residence</gml:LocationString>
</gml:location>
```

Location given by another service:

```
<gml:location xlink:href="http://www.ga.gov.au/bin/gazm01?placename=leederville&placetype=R&state=WA+/">
```

### 8.2.2.3 gml:priorityLocation, gml:priorityLocationType

A property gml:priorityLocation is provided for GML Application Schema developers that wish to provide prioritized locations for their features. A gml:priorityLocation has the following content model:

```
<element name="priorityLocation" type="gml:PriorityLocationPropertyType"
  substitutionGroup="gml:location"/>
```

```

<complexType name="PriorityLocationPropertyType">
  <complexContent>
    <extension base="gml:LocationPropertyType">
      <attribute name="priority" type="string" use="optional"/>
    </extension>
  </complexContent>
</complexType>

```

Note that this simply adds a priority string to the base gml:location property. This can be used to assign levels of importance to the different locations (e.g. location by location string, geometry etc.).

**NOTE** The element gml:priorityLocation is deprecated in this version of GML.

#### 8.2.2.4 gml:featureProperty, gml:Feature.PropertyType, gml:featureMember

A particular class of properties define associations between features. These use the gml:AssociationType pattern as follows:

```

<complexType name="Feature.PropertyType">
  <sequence>
    <element ref="gml:_Feature" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

```

The concrete elements gml:featureMember and gml:featureProperty use this content model, and are declared as follows:

```

<element name="featureProperty" type="gml:Feature.PropertyType"/>
<element name="featureMember" type="gml:Feature.PropertyType"/>

```

#### 8.2.2.5 gml:featureMembers, gml:FeatureArray.PropertyType

At times it is useful to define a property containing an array of other features. This is done using a feature array property type as defined by the following content model:

```

<complexType name="FeatureArray.PropertyType">
  <sequence>
    <element ref="gml:_Feature" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>

```

The concrete elements gml:featureMembers uses this content model, and is declared as follows:

```
<element name="featureMembers" type="gml:FeatureArray.PropertyType"/>
```

### 8.2.3 Geometry Properties

In general the definition of feature properties is in the responsibility of the application schema designer. However, since the OGC abstract specification defines a limited set of basic geometry types, GML defines a set of predefined geometric property elements to associate instances of these geometry types with features.

The GML Feature schema also provides descriptive names for the geometry properties, encoded as common English language terms. Overall, there are three levels of naming geometry properties in GML:

**Formal names** that denote geometry properties in a manner based on the type of geometry allowed as a property value. These are names based on the name of the geometry type with a suffix "Property".

**Table 1 Pre-defined formal geometry properties**

<b>Formal property name</b>	<b>XML Schema property type</b>	<b>Associated geometry object types (element names)</b>
pointProperty	PointPropertyType	Point
curveProperty	CurvePropertyType	_Curve LineString Curve OrientableCurve CompositeCurve
surfaceProperty	SurfacePropertyType	_Surface Polygon Surface OrientableSurface CompositeSurface
solidProperty	SolidPropertyType	_Solid Solid CompositeSolid
multiPointProperty	MultiPointPropertyType	MultiPoint
multiCurveProperty	MultiCurvePropertyType	MultiCurve
multiSurfaceProperty	MultiSurfacePropertyType	MultiSurface
multiSolidProperty	MultiSolidPropertyType	MultiSolid
multiGeometryProperty	MultiGeometryPropertyType	MultiGeometry
pointArrayProperty	PointArrayPropertyType	Point(s)
curveArrayProperty	CurveArrayPropertyType	_Curve(s) LineString(s) Curve(s) OrientableCurve(s) CompositeCurve(s)
surfaceArrayProperty	SurfaceArrayPropertyType	_Surface(s) Polygon(s) Surface(s) OrientableSurface(s) CompositeSurface(s)
solidArrayProperty	SolidArrayPropertyType	_Solid(s) Solid(s) CompositeSolid(s)

The precise semantics of these geometry properties (e.g. "What does pointProperty of an object mean?") is not specified.

**Descriptive names** that provide a set of standardized synonyms or aliases defined in feature.xsd for the formal names; these allow use of a more user-friendly set of terms. These are:

```

<element name="centerOf" type="gml:PointPropertyType"/>
<element name="position" type="gml:PointPropertyType"/>
<element name="extentOf" type="gml:SurfacePropertyType"/>
<element name="edgeOf" type="gml:CurvePropertyType"/>
<element name="centerLineOf" type="gml:CurvePropertyType"/>

<element name="multiLocation" type="gml:MultiPointPropertyType"/>
<element name="multiCenterOf" type="gml:MultiPointPropertyType"/>
<element name="multiPosition" type="gml:MultiPointPropertyType"/>
<element name="multiCenterLineOf" type="gml:MultiCurvePropertyType"/>
<element name="multiEdgeOf" type="gml:MultiCurvePropertyType"/>
<element name="multiCoverage" type="gml:MultiSurfacePropertyType"/>
<element name="multiExtentOf" type="gml:MultiSurfacePropertyType"/>

```

These property elements provide common role names for the geometry of geographic features. The specific semantics of these role names (e.g. centerOf) is not defined in GML, and it is expected that additional clarification would be provided in <annotations> within the using GML application schema.

**Application-specific names** chosen by users and defined in application schemas based on GML.

There are no inherent restrictions in the type of geometry property a feature type may have. For example, a RadioTower feature type could have a *location* that returns a Point geometry to identify its location, and have another geometry property called *extentOf* that returns a Polygon geometry describing its physical structure, and have yet a third geometry property called *serviceArea* that returns a Polygon geometry describing the area in which its transmissions can be received reliably.

#### 8.2.4 Topology Properties

In general the definition of topology properties is in the responsibility of the application schema designer. However, GML defines topology property elements that may be used to associate instances of topology types with features. All of these predefined topology property elements have formal names, so there are just two levels of naming topology properties in GML:

**Formal names** that denote geometry properties in a manner based on the type of topology allowed as a property value. Note that the first four of these properties express direction, whereas the others do not.

**Table 2 Pre-defined formal topology properties**

Formal property name	XML Schema property type	Associated topology object types (element names)
directedNode	DirectedNodePropertyType	Node
directedEdge	DirectedEdgePropertyType	Edge
directedFace	DirectedFacePropertyType	Face
directedTopoSolid	DirectedTopoSolidPropertyType	TopoSolid
topoPointProperty	TopoPointPropertyType	TopoPoint
topoCurveProperty	TopoCurvePropertyType	TopoCurve
topoSurfaceProperty	TopoSurfacePropertyType	TopoSurface
topoVolumeProperty	TopoVolumePropertyType	TopoVolume
topoComplexProperty	TopoComplexMemberType	TopoComplex

**Application-specific names** chosen by users and defined in application schemas based on GML.

There are no inherent restrictions of the type of topology property a feature type may have. For example, a StatisticalArea feature type could have one or more topoCurveProperty that returns a TopoCurve to represent

the boundary of the Statistical Area, and one or more topoSurfaceProperty that returns a TopoSurface to represent the Statistical Area itself.

## 8.2.5 Temporal Properties

### 8.2.5.1 General purpose temporal properties

GML includes a small number of temporal property types that may be used in application schemas. Four (concrete) global elements representing time properties are provided for general use (compare with geometry properties described in clause 8.2.3):

gml:timePosition - for associating a direct time position with a feature or other object, encoded as a literal

gml:duration, gml:timeInterval - for associating a duration with a feature or other object

gml:validTime - for associating a time primitive with a feature or other object.

However, the temporal objects and the property types that refer to them provide a relatively comprehensive set of components for associating temporal information with features and other objects.

### 8.2.5.2 Specific temporal properties

In general the definition of feature properties is in the responsibility of the application schema designer. Following the standard GML property model, the name of a property element should reflect the semantics of the relationship of the temporal object with the containing feature or other object. For example a feature type may have a *constructionTime* property whose XML type is “gml:TimePeriodPropertyType”, a *completionTime* property whose XML type is “gml:TimeInstantPropertyType”, and an *age* whose XML type is “gml:TimeDurationType” or “gml:TimeIntervalLengthType”. The following types are provided for direct use in declaring property elements:

XML Schema property type	Associated object types and substitutable object types
TimePrimitivePropertyType	TimePrimitive, TimeGeometricPrimitive, TimeInstant, TimePeriod, TimeTopologyPrimitive, TimeEdge, TimeNode
TimeGeometricPrimitivePropertyType	TimeGeometricPrimitive, TimeInstant, TimePeriod
TimeInstantPropertyType	TimeInstant
TimePeriodPropertyType	TimePeriod
TimeTopologyPrimitivePropertyType	TimeTopologyPrimitive, TimeEdge, TimeNode
TimeEdgePropertyType	TimeEdge
TimeNodePropertyType	TimeNode
TimeTopologyComplexPropertyType	TimeTopologyComplex
TimeOrdinalEraPropertyType	TimeOrdinalEra
TimeCalendarPropertyType	TimeCalendar
TimeCalendarEraPropertyType	TimeCalendarEra

TimeClock.PropertyType	TimeClock
------------------------	-----------

## 8.2.6 Defining specific feature types

All specific feature types defined in application schemas shall be derived from `gml:AbstractFeatureType`, and thus all GML features inherit the optional `gml:boundedBy` and `gml:location` properties, as well as the standard `gml:metaDataProperty`, `gml:description` and `gml:name` properties inherited in turn from `gml:AbstractGMLType`, unless any property is suppressed in a derivation by restriction. `AbstractFeatureType` also inherits `gml:id` from `gml:AbstractGMLType` and this is the preferred means of supporting database identifiers in GML 3.

This type derivation requirement means that general purpose software designed to process arbitrary GML data shall be able to traverse the XML Schema derivation tree in order to determine whether or not a given element in the data stream is a GML feature.

A GML feature has a set of properties, where the specific set of properties defines the feature type. This follows the general definition of a feature given in ISO 19109, 19110 and the OGC Abstract Specification Topic 5. Properties have simple values, using XML Schema simpleContent types, or properties may have complex values, in which case they should be declared using the `gml:AssociationType` pattern described in Clause 7.5.3.

In the application schema defining a feature there shall be a global element declared whose name is the semantic type of the feature in the domain of discourse.

```
<element name="<<featureName>>" type = "<<contentModel >>" />
```

If that feature element may be used in a general `gml:FeatureCollection`, or in a feature collection defined by a different application schema, then its global element declaration shall be made a member of the `gml:_Feature` substitution group:

```
<element name="<<featureName>>" type = "<<contentModel >>" substitutionGroup="gml:_Feature" />
```

The content model of the feature may be a named (or anonymous) complex type.

### 8.2.6.1 gml:BoundedFeatureType

A simple restriction of `gml:AbstractFeatureType` makes the optional `boundedBy` property mandatory. `gml:BoundedFeatureType` is defined as follows:

```
<complexType name="BoundedFeatureType" abstract="true">
  <complexContent>
    <restriction base="gml:AbstractFeatureType">
      <sequence>
        <group ref="gml:StandardObjectProperties"/>
        <element ref="gml:boundedBy"/>
        <element ref="gml:location" minOccurs="0">
          <annotation>
            <appinfo>deprecated</appinfo>
            <documentation>deprecated in GML version 3.1</documentation>
          </annotation>
        </element>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
```

NOTE In XML Schema a content model derived by restriction must lead to a content model which is valid according to the base type – i.e. it is a valid subset of the parent. In the schema document this is expressed by copying down all the XML elements that are required for the content of the derived type, with their occurrence counts. In this instance the

adjustment made in the derived type is that an element that was optional in the parent (gml:boundedBy had minOccurs="0") is made mandatory (no occurrence constraints implies exactly one). Any element from the parent that is omitted (which can only be those with minOccurs="0" in the parent definition) is not inherited by the derived type. In contrast, it is not necessary to copy down the XML attributes, only those which are altered by restriction need be mentioned. By default the derived type has the same XML attributes and occurrence constraints as the parent. See XML Schema Part 1 (Structures) for more detail.

## 8.2.7 Feature Collections

A GML Feature Collection is a list of GML feature instances.

### 8.2.7.1 gml:AbstractFeatureCollectionType

GML feature collections shall be derived by extension or restriction from gml:AbstractFeatureCollectionType, defined as follows:

```
<complexType name="AbstractFeatureCollectionType" abstract="true">
<complexContent>
<extension base="gml:AbstractFeatureType">
<sequence>
<element ref="gml:featureMember" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:featureMembers" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>
```

The gml:featureMember property (but not the gml:featureMembers property) follows the association pattern and can thus refer to a "remote" feature by means of the xlink:ref attribute.

The compositing property gml:featureMembers encloses a set of members of the Feature Collection regardless of their semantic type as features. gml:featureMember encloses or references a single feature instance. gml:featureMember and gml:featureMembers properties can appear on the same Feature Collection, but there can be only one gml:featureMembers property.

GML Feature Collections are themselves valid GML features and can have gml:location and other properties as defined in their GML Application Schema (see chapter 8.).

### 8.2.7.2 gml:\_FeatureCollection, gml:FeatureCollection, gml:FeatureCollectionType

```
<element name="_FeatureCollection" type="gml:AbstractFeatureCollectionType" abstract="true"
substitutionGroup="gml:_Feature"/>
```

This abstract element gml:\_FeatureCollection serves as the head of a substitution group which may contain any elements whose content model is derived from gml:AbstractFeatureType. This may be used as a variable in the construction of content models.

The schema also provides a concrete feature collection:

```
<element name="FeatureCollection" type="gml:FeatureCollectionType" substitutionGroup="gml:_Feature"/>

<complexType name="FeatureCollectionType">
<complexContent>
<extension base="gml:AbstractFeatureCollectionType"/>
</complexContent>
</complexType>
```

Users of the concrete gml:FeatureCollection should note that it allows any valid GML feature as a member.

The content model of a GML Feature Collection shall be derived from `gml:AbstractFeatureCollectionType`. This in turn derives from `gml:AbstractFeatureType`. Hence feature collections are features, and are in general substitutable for `gml:_Feature`.

### 8.3 Spatial reference system used in a feature or feature collection

The value of the `gml:boundedBy` property for a feature or feature collection is usually a `gml:Envelope`. In common with all geometry elements derived from `gml:AbstractGeometryType` (clause 9.1.2.1), the coordinate reference system used for the positions defining the `gml:Envelope` may be indicated using the optional XML attribute “`srsName`”.

For convenience in constructing feature and feature collection instances, the value of the `srsName` attribute on the `gml:Envelope` (or `gml:Box`) which is the value of the `gml:boundedBy` property of the feature shall be inherited by all directly expressed geometries in all properties of the feature or members of the collection, unless overruled by the presence of a local `srsName`. Thus it is not necessary for a geometry to carry an `srsName` attribute if it uses the same coordinate reference system as given on the `gml:boundedBy` property of its parent feature. Inheritance of the coordinate reference system continues to any depth of nesting, but if overruled by a local `srsName` declaration, then the new coordinate reference system is inherited by all its children in turn.

Notwithstanding this rule, all the geometries used in a feature or feature collection may carry `srsName` attributes, in order to indicate the reference system used locally, even if they are the same as the parent.

## 9 GML schemas – basic geometry

### 9.1 General Concepts

#### 9.1.1 Overview

NOTE The geometry model of GML complies with ISO 19107. The underlying concepts of the types and elements of the GML geometry model are discussed in this document in more detail.

This clause describes the basic geometry model used by GML in the XML Schema documents:

- `geometryBasic0d1d.xsd`
- `geometryBasic2d.xsd`

The basic GML types and elements are described in the schemas listed in Annex C. The geometry schemas are identified by the following location-independent name (using URN syntax):

`urn:opengis:specification:gml:schema-xsd:geometryBasic0d1d:v3.1.0`  
`urn:opengis:specification:gml:schema-xsd:geometryBasic2d:v3.1.0`

All types, elements and attributes in these XML Schema documents are covered by this Clause.

NOTE The GML geometry model is complex. To make the schemas more accessible to readers with or without prior experience with GML 2, the different types and elements are grouped in several XML Schema documents.

Any geometry element that inherits the semantics of `AbstractGeometryType` can be viewed as a set of direct positions.

All of the classes derived from `AbstractGeometryType` inherit an optional association to a coordinate reference system. All direct positions shall directly or indirectly be associated with a coordinate reference system. When geometry elements are aggregated in another geometry element (such as a `MultiGeometry` or `GeometricComplex`), which already has a coordinate reference system specified, then these elements are assumed to be in that same coordinate reference system unless otherwise specified.

The geometry model distinguishes geometric primitives, aggregates and complexes.

Geometric primitives (i.e. instances of a subtype of AbstractGeometricPrimitiveType) will be open, that is, they will not contain their boundary points; curves will not contain their end points, surfaces will not contain their boundary curves, and solids will not contain their bounding surfaces.

## 9.1.2 Abstract Geometry

### 9.1.2.1 **gml:AbstractGeometryType**

```
<complexType name="AbstractGeometryType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <attribute name="gid" type="string" use="optional"/>
      <annotation>
        <appinfo>deprecated</appinfo>
        <documentation>deprecated in GML version 3.0</documentation>
      </annotation>
    </attribute>
    <attributeGroup ref="gml:SRSReferenceGroup"/>
  </extension>
</complexContent>
</complexType>
```

All geometry elements are derived directly or indirectly from this abstract supertype. A geometry element may have an identifying attribute ("gml:id"), a name (attribute "name") and a description (attribute "description"). It may be associated with a spatial reference system (attribute group "gml:SRSReferenceGroup").

The following rules shall be adhered to:

- Every geometry type shall derive from this abstract type.
- Every geometry element (i.e. an element of a geometry type) shall be directly or indirectly in the substitution group of \_Geometry.

*NOTE The attribute "gid" is included for backward compatibility with GML 2 and is deprecated with GML 3. This identifier is superceded by "gml:id" inherited from AbstractGMLType. The attribute "gid" should not be used anymore and may be deleted in future versions of GML without further notice.*

### 9.1.2.2 **SRSReferenceGroup**

```
<attributeGroup name="SRSReferenceGroup">
  <attribute name="srsName" type="anyURI" use="optional"/>
  <attribute name="srsDimension" type="positiveInteger" use="optional"/>
  <attributeGroup ref="gml:SRSInformationGroup"/>
</attributeGroup>
```

The attribute group "SRSReferenceGroup" is an optional reference to the CRS used by this geometry, with optional additional information to simplify the processing of the coordinates when a more complete definition of the CRS is not needed.

In general the attribute "srsName" points to a CRS instance of gml:CoordinateReferenceSystemType (see coordinateReferenceSystems.xsd). For well-known references it is not required that the CRS description exists at the location the URI points to.

If no srsName attribute is given, the CRS shall be specified as part of the larger context this geometry element is part of, e.g. a geometric aggregate.

**NOTE** The name “srsName” has been chosen deliberately. In the current version of GML “crsName” would be more appropriate, however, in future versions other types of spatial reference systems, i.e. those using geographic identifiers, may be supported by GML, too.

The optional attribute “srsDimension” is the length of coordinate sequence in a position. This dimension is a derived from the coordinate reference system. When the “srsName” attribute is omitted, this attribute shall be omitted.

#### 9.1.2.3 SRSInformationGroup

```
<attributeGroup name="SRSInformationGroup">
  <attribute name="axisLabels" type="gml:NCNameList" use="optional"/>
  <attribute name="uomLabels" type="gml:NCNameList" use="optional"/>
</attributeGroup>
```

The attributes “uomLabels” and “axisLabels”, defined in the “SRSInformationGroup” attribute group, are optional additional and redundant information for a CRS to simplify the processing of the coordinate values when a more complete definition of the CRS is not needed. This information shall be the same as included in the complete definition of the CRS, referenced by the “srsName” attribute. When the “srsName” attribute is included, either both or neither of the “axisLabels” and “uomLabels” attributes shall be included. When the “srsName” attribute is omitted, both of these attributes shall be omitted.

The attribute “axisLabels” is an ordered list of labels for all the axes of this CRS. The “gml:axisAbbrev” value should be used for these axis labels, after spaces and forbidden characters are removed. When the “srsName” attribute is included, this attribute is optional. When the “srsName” attribute is omitted, this attribute shall also be omitted.

The attribute “uomLabels” is an ordered list of unit of measure (uom) labels for all the axes of this CRS. The value of the string in the “gml:catalogSymbol” should be used for this uom labels, after spaces and forbidden characters are removed. When the “axisLabels” attribute is included, this attribute shall also be included. When the “axisLabels” attribute is omitted, this attribute shall also be omitted.

#### 9.1.2.4 gml:\_Geometry

```
<element name="_Geometry" type="gml:AbstractGeometryType" abstract="true" substitutionGroup="gml:_GML" />
```

The “\_Geometry” element is the abstract head of the substitution group for all geometry elements of GML 3. This includes pre-defined and user-defined geometry elements. Any geometry element shall be a direct or indirect extension/restriction of AbstractGeometryType and shall be directly or indirectly in the substitution group of “\_Geometry”.

#### 9.1.2.5 gml:Geometry.PropertyType

```
<complexType name="Geometry.PropertyType">
  <sequence>
    <element ref="gml:_Geometry" minOccurs="0" />
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

A geometric property can either be any geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same or another document). Note that either the reference or the contained element shall be given, but not both or none. See clause 7.5.3.

If a feature has a property, which takes a geometry element as its value, this is called a geometry property. A generic type for such a geometry property is Geometry.PropertyType which follows the general rules described in clauses 7.5.3.

#### 9.1.2.6 gml:GeometryArrayPropertyType

```
<complexType name="GeometryArrayPropertyType">
  <sequence>
    <element ref="gml:_Geometry" minOccurs="0" maxOccurs="unbounded" />
  </sequence>
</complexType>
```

A container for an array of geometry elements. The elements are always contained in the array property, referencing geometry elements or arrays of geometry elements via XLinks is not supported.

If a feature has a property which takes an array of geometry elements as its value, this is called a geometry array property. A generic type for such a geometry property is GeometryArray.PropertyType which follows the general rules described in clause 7.5.3.

In general the elements in an array have to be homogenous, containing elements that are all of the same type. For example, all elements in a GeometryArray.PropertyType are of the type AbstractGeometryType (including types derived from this abstract base type) as long as the element is directly or indirectly substitutable for gml:\_Geometry.

#### 9.1.3 Coordinate Geometry

##### 9.1.3.1 gml:DirectPositionType, gml:pos

```
<complexType name="DirectPositionType">
  <simpleContent>
    <extension base="gml:doubleList">
      <attributeGroup ref="gml:SRSReferenceGroup"/>
    </extension>
  </simpleContent>
</complexType>

<element name="pos" type="gml:DirectPositionType" />
```

DirectPosition instances hold the coordinates for a position within some coordinate reference system (CRS). Since DirectPositions, as data types, will often be included in larger objects (such as geometry elements) that have references to CRS, the "srsName" attribute will in general be missing, if this particular DirectPosition is included in a larger element with such a reference to a CRS. In this case, the CRS is implicitly assumed to take on the value of the containing object's CRS.

The attribute group "SRSReferenceGroup" is described in clause 9.1.2.2. If no srsName attribute is given, the CRS shall be specified as part of the larger context this geometry element is part of, e.g. a geometric element like point, curve, etc. It is expected that the attribute will be specified at the direct position level only in rare cases.

##### 9.1.3.2 gml:DirectPositionListType, gml:posList

```
<complexType name="DirectPositionListType">
  <simpleContent>
    <extension base="gml:doubleList">
      <attributeGroup ref="gml:SRSReferenceGroup"/>
      <attribute name="count" type="positiveInteger" use="optional" />
    </extension>
  </simpleContent>
</complexType>
```

```
<element name="posList" type="gml:DirectPositionListType" />
```

posLists instances (and other instances with the content model specified by DirectPositionListType) hold the coordinates for a sequence of direct positions within the same coordinate reference system (CRS).

The attribute group “SRSReferenceGroup” is described in clause 9.1.2.2. If no srsName attribute is given, the CRS shall be specified as part of the larger context this geometry element is part of, e.g. a geometric element like point, curve, etc. It is expected that the attribute will be specified at the direct position level only in rare cases.

The optional attribute “count” allows to specify the number of direct positions in the list. If the attribute “count” is present then the attribute “srsDimension” shall be present, too.

The number of entries in the list is equal to the product of the dimensionality of the coordinate reference system (i.e. it is a derived value of the coordinate reference system definition) and the number of direct positions.

### 9.1.3.3 gml:geometricPositionGroup

```
<group name="geometricPositionGroup">
  <choice>
    <element ref="gml:pos"/>
    <element ref="gml:pointProperty"/>
  </choice>
</group>
```

GML supports two different ways to specify a geometric position: either by a direct position (a data type) or a point (a geometric object).

“pos” elements are positions that are “owned” by the geometric primitive encapsulating this geometric position.

“pointProperty” elements contain a point that may be referenced from other geometry elements or reference another point defined elsewhere (reuse of existing points).

### 9.1.3.4 gml:geometricPositionListGroup

```
<group name="geometricPositionListGroup">
  <choice>
    <element ref="gml:posList"/>
    <group ref="gml:geometricPositionGroup" maxOccurs="unbounded"/>
  </choice>
</group>
```

GML supports two different ways to specify a list of geometric positions: either by a sequence of geometric positions (by reusing the group definition) or a sequence of direct positions (element “posList”).

The “posList” element allows for a compact way to specify the coordinates of the positions, if all positions are represented in the same coordinate reference systems.

**NOTE** The definition of this group may be used as a pattern in the definition of geometric primitives instead of using this group definition directly. The main change will typically be a change in the multiplicity of the referenced group. A LineString, for example, requires at least two positions.

### 9.1.3.5 gml:CoordinatesType, gml:coordinates

```
<complexType name="CoordinatesType">
  <simpleContent>
    <extension base="string">
```

```

<attribute name="decimal" type="string" default=". "/>
<attribute name="cs" type="string" default=","/>
<attribute name="ts" type="string" default="\u0020;" />
</extension>
</simpleContent>
</complexType>

<element name="coordinates" type="gml:CoordinatesType" />

```

This type is deprecated with GML version 3.1.0 for tuples with ordinate values that are numbers.

CoordinatesType is a text string, intended to be used to record an array of tuples or coordinates.

*While it is not possible to enforce the internal structure of the string through schema validation, some optional attributes have been provided in previous versions of GML to support a description of the internal structure. These attributes are deprecated with GML version 3.1.0. The attributes are intended to be used as follows:*

**decimal** symbol used for a decimal point  
(default=“.” a stop or period)

**cs** symbol used to separate components within a tuple or coordinate string  
(default=“,” a comma)

**ts** symbol used to separate tuples or coordinate strings  
(default=“ ” a space)

The “coordinates” element is deprecated with GML version 3.1.0.

Since it is based on the XML Schema string type, CoordinatesType can be used in the construction of tables of tuples or arrays of tuples, including ones that contain mixed text and numeric values. For example:

```
<my:tupleList>bettong,357.,2.3 skink,140.,0.75 wombat,770.,17.5</my:tupleList>
```

#### 9.1.3.6 gml:CoordType, gml:coord

```

<complexType name="CoordType">
  <sequence>
    <element name="X" type="decimal" />
    <element name="Y" type="decimal" minOccurs="0" />
    <element name="Z" type="decimal" minOccurs="0" />
  </sequence>
</complexType>

```

Represents a coordinate tuple in one, two, or three dimensions. Deprecated with GML 3.0 and replaced by DirectPositionType.

```
<element name="coord" type="gml:CoordType" />
```

Deprecated with GML 3.0 and included for backwards compatibility with GML 2. Use the “pos” element instead.

### 9.1.4 Vectors

#### 9.1.4.1 gml:VectorType, gml:Vector

```

<complexType name="VectorType">
  <simpleContent>
    <restriction base="gml:DirectPositionType"/>
  </simpleContent>
</complexType>

```

```
<element name="vector" type="gml:VectorType" />
```

A vector is an ordered set of numbers called coordinate that represent a position in space that is defined by a coordinate reference system (CRS). For some application the components of the position may be adjusted to yield a unit vector.

NOTE This definition allows VectorType to be used elsewhere when appropriate – e.g. for offsetVector in grids.xsd, and vector to be used directly when appropriate – e.g. in DirectionVector in direction.xsd.

### 9.1.5 Envelopes

#### 9.1.5.1 gml:EnvelopeType, gml:Envelope

```
<complexType name="EnvelopeType">
  <choice>
    <sequence>
      <element name="lowerCorner" type="gml:DirectPositionType"/>
      <element name="upperCorner" type="gml:DirectPositionType"/>
    </sequence>
    <element ref="gml:coord" minOccurs="2" maxOccurs="2"/>
    <element ref="gml:pos" minOccurs="2" maxOccurs="2"/>
    <element ref="gml:coordinates"/>
  </choice>
  <attributeGroup ref="gml:SRSReferenceGroup"/>
</complexType>

<element name="Envelope" type="gml:EnvelopeType"/>
```

Envelope defines an extent using a pair of positions defining opposite corners in arbitrary dimensions. The first direct position is the "lower corner" (a coordinate position consisting of all the minimal ordinates for each dimension for all points within the envelope), the second one the "upper corner" (a coordinate position consisting of all the maximal ordinates for each dimension for all points within the envelope).

*The use of the properties "coord", "coordinates" and "pos" has been deprecated. The explicitly named properties "lowerCorner" and "upperCorner" shall be used instead.*

NOTE Regardless of dimension, an envelope can be represented without ambiguity as two direct positions (coordinate points) provided the ordering of those points adheres to the specified rule. Envelope is often referred to as a minimum bounding box or rectangle. However, this Envelope will not always specify the MINIMUM rectangular bounding region, if the referenced CRS is a Geographic CRS, or uses an Ellipsoidal, Spherical, Polar, or Cylindrical coordinate system, as those terms are specified in Subclause 12.4. Specifically, this Envelope will not specify the MINIMUM rectangular bounding region of a geometry whose set of points span the value discontinuity in an angular coordinate axis. Such axes include the Longitude and Latitude of Ellipsoidal and Spherical coordinate systems. That geometry could lie within a small region on the surface of the ellipsoid or sphere, or could extend completely around the ellipsoid or sphere.

## 9.2 Geometric Primitives

### 9.2.1 Simple Geometric Primitives (0- and 1-dimensional)

#### 9.2.1.1 gml:AbstractGeometricPrimitiveType, gml:\_GeometricPrimitive

```
<complexType name="AbstractGeometricPrimitiveType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractGeometryType" />
  </complexContent>
</complexType>
```

This is the abstract root type of the geometric primitives. A geometric primitive is a geometric object that is not decomposed further into other primitives in the system. All primitives are oriented in the direction implied by the sequence of their coordinate tuples.

```
<element name="_GeometricPrimitive" type="gml:AbstractGeometricPrimitiveType" abstract="true"
substitutionGroup="gml:_Geometry" />
```

The "\_GeometricPrimitive" element is the abstract head of the substitution group for all (pre- and user-defined) geometric primitives.

```
<complexType name="GeometricPrimitivePropertyType">
  <sequence>
    <element ref="gml:_GeometricPrimitive" minOccurs="0" />
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

A property that has a geometric primitive as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

#### 9.2.1.2 gml:PointType, gml:Point

```
<complexType name="PointType">
  <complexContent>
    <extension base="gml:AbstractGeometricPrimitiveType">
      <sequence>
        <choice>
          <element ref="gml:pos" />
          <element ref="gml:coordinates" />
          <element ref="gml:coord">
            <annotation>
              <appinfo>deprecated</appinfo>
              <documentation>deprecated in GML version 3.0</documentation>
            </annotation>
          </element>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="Point" type="gml:PointType" substitutionGroup="gml:_GeometricPrimitive" />
```

A Point is defined by a single coordinate tuple. The direct position of a point is specified by the "pos" element which is of type DirectPositionType.

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### 9.2.1.3 gml:PointPropertyType, gml:pointProperty

```
<complexType name="PointPropertyType">
  <sequence>
    <element ref="gml:Point" minOccurs="0" />
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

A property that has a point as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

```
<element name="pointProperty" type="gml:PointPropertyType" />
```

This property element either references a point via the XLink-attributes or contains the point element. pointProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for Point.

#### 9.2.1.4 gml:PointArrayPropertyType, gml:pointArrayProperty

```
<complexType name="PointArrayPropertyType">
  <sequence>
    <element ref="gml:Point" minOccurs="0" maxOccurs="unbounded" />
  </sequence>
</complexType>
```

A container for an array of points. The elements are always contained in the array property, referencing geometry elements or arrays of geometry elements via XLinks is not supported.

```
<element name="pointArrayProperty" type="gml:PointArrayPropertyType" />
```

This property element contains a list of point elements. pointArrayProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for a list of Points.

#### 9.2.1.5 gml:AbstractCurveType, gml:\_Curve

```
<complexType name="AbstractCurveType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractGeometricPrimitiveType" />
  </complexContent>
</complexType>
```

An abstraction of a curve to support the different levels of complexity. The curve can always be viewed as a geometric primitive, i.e. is continuous.

```
<element name="_Curve" type="gml:AbstractCurveType" abstract="true"
substitutionGroup="gml:_GeometricPrimitive" />
```

The "\_Curve" element is the abstract head of the substitution group for all (continuous) curve elements.

#### 9.2.1.6 gml:Curve.PropertyType, gml:curveProperty

```
<complexType name="Curve.PropertyType">
  <sequence>
    <element ref="gml:_Curve" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

A property that has a curve as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

```
<element name="curveProperty" type="gml:Curve.PropertyType" />
```

This property element either references a curve via the XLink-attributes or contains the curve element. curveProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for \_Curve.

### 9.2.1.7 gml:CurveArrayPropertyType, gml:curveArrayProperty

```
<complexType name="CurveArrayPropertyType">
  <sequence>
    <element ref="gml:_Curve" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
```

A container for an array of curves. The elements are always contained in the array property, referencing geometry elements or arrays of geometry elements via XLinks is not supported.

```
<element name="curveArrayProperty" type="gml:CurveArrayPropertyType" />
```

This property element contains a list of curve elements. curveArrayProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for a list of \_Curves.

### 9.2.1.8 gml:LineStringType, gml:LineString

```
<complexType name="LineStringType">
  <complexContent>
    <extension base="gml:AbstractCurveType">
      <sequence>
        <choice>
          <choice minOccurs="2" maxOccurs="unbounded">
            <element ref="gml:pos" />
            <element ref="gml:pointProperty" />
            <element ref="gml:coord">
              <annotation>
                <appinfo>deprecated</appinfo>
                <documentation>deprecated in GML version 3.0</documentation>
              </annotation>
            </element>
          </choice>
          <element ref="gml:posList" />
          <element ref="gml:coordinates" />
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="LineString" type="gml:LineStringType" substitutionGroup="gml:_Curve" />
```

A LineString is a special curve that consists of a single segment with linear interpolation. It is defined by two or more coordinate tuples, with linear interpolation between them. It is backwards compatible with the LineString of GML 2. GM\_LineString of ISO 19107 is implemented by LineStringSegment.

GML supports two different ways to specify the control points of a line string:

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve (reuse of existing points).
- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve only. The number of direct positions in the coordinates list shall be at least two and include the start and end point. Note that the start and end points of one line string are often identical with those of other line strings.

*The use of the element “coordinates” is deprecated with GML version 3.1.0. Use “posList” instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### **9.2.1.9    *gml:LineStringPropertyType, gml:lineStringProperty***

```
<complexType name="LineStringPropertyType">
  <sequence>
    <element ref="gml:LineString" minOccurs="0" />
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

*This type is deprecated with GML 3 and shall not be used. It is included for backwards compatibility with GML 2. Use CurvePropertyType instead.*

*A property that has a line string as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.*

```
<element name="lineStringProperty" type="gml:LineStringPropertyType" />
```

*Deprecated with GML 3 and included only for backwards compatibility with GML 2. Use "curveProperty" instead.*

*This property element either references a line string via the XLink-attributes or contains the line string element.*

### **9.2.2    Simple Geometric Primitives (2-dimensional)**

#### **9.2.2.1    *gml:AbstractSurfaceType, gml:\_Surface***

```
<complexType name="AbstractSurfaceType">
  <complexContent>
    <extension base="gml:AbstractGeometricPrimitiveType" />
  </complexContent>
</complexType>
```

An abstraction of a surface to support the different levels of complexity. A surface is always a continuous region of a plane.

```
<element name="_Surface" type="gml:AbstractSurfaceType" abstract="true"
substitutionGroup="gml:_GeometricPrimitive" />
```

The "\_Surface" element is the abstract head of the substitution group for all (continuous) surface elements.

#### **9.2.2.2    *gml:SurfacePropertyType, gml:surfaceProperty***

```
<complexType name="SurfacePropertyType">
  <sequence>
    <element ref="gml:_Surface" minOccurs="0" />
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

*A property that has a surface as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.*

```
<element name="surfaceProperty" type="gml:SurfacePropertyType" />
```

This property element either references a surface via the XLink-attributes or contains the surface element. surfaceProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for \_Surface.

### 9.2.2.3 gml:SurfaceArrayPropertyType, gml:surfaceArrayProperty

```
<complexType name="SurfaceArrayPropertyType">
  <sequence>
    <element ref="gml:_Surface" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
```

A container for an array of surfaces. The elements are always contained in the array property, referencing geometry elements or arrays of geometry elements via XLinks is not supported.

```
<element name="surfaceArrayProperty" type="gml:SurfaceArrayPropertyType" />
```

This property element contains a list of surface elements. surfaceArrayProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for a list of \_Surfaces.

### 9.2.2.4 gml:PolygonType, gml:Polygon

```
<complexType name="PolygonType">
  <complexContent>
    <extension base="gml:AbstractSurfaceType">
      <sequence>
        <element ref="gml:exterior" minOccurs="0" />
        <element ref="gml:interior" minOccurs="0" maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="Polygon" type="gml:PolygonType" substitutionGroup="gml:_Surface" />
```

A Polygon is a special surface that is defined by a single surface patch. The boundary of this patch is coplanar and the polygon uses planar interpolation in its interior. It is backwards compatible with the Polygon of GML 2.

NOTE GM\_Polygon of ISO 19107 is implemented by PolygonPatch.

The elements "exterior" and "interior" describe the surface boundary of the polygon and are specified below.

### 9.2.2.5 gml:exterior, gml:interior, gml:outerBoundaryIs, gml:innerBoundaryIs

```
<element name="exterior" type="gml:AbstractRingPropertyType" />
```

A boundary of a surface consists of a number of rings. In the normal 2D case, one of these rings is distinguished as being the exterior boundary. In a general manifold this is not always possible, in which case all boundaries shall be listed as interior boundaries, and the exterior will be empty.

```
<element name="interior" type="gml:AbstractRingPropertyType" />
```

A boundary of a surface consists of a number of rings. The "interior" rings separate the surface / surface patch from the area enclosed by the rings.

```
<element name="outerBoundaryIs" type="gml:AbstractRingPropertyType" substitutionGroup="gml:exterior" />
```

*Deprecated with GML 3, included only for backwards compatibility with GML 2. Use "exterior" instead.*

```
<element name="innerBoundaryIs" type="gml:AbstractRingPropertyType" substitutionGroup="gml:interior" />
```

Deprecated with GML 3, included only for backwards compatibility with GML 2. Use "interior" instead.

#### 9.2.2.6 gml:AbstractRingType, gml:\_Ring

```
<complexType name="AbstractRingType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractGeometryType" />
  </complexContent>
</complexType>
```

An abstraction of a ring to support surface boundaries of different complexity.

```
<element name="_Ring" type="gml:AbstractRingType" abstract="true" />
```

The "\_Ring" element is the abstract head of the substitution group for all closed boundaries of a surface patch.

#### 9.2.2.7 gml:AbstractRingPropertyType

```
<complexType name="AbstractRingPropertyType">
  <sequence>
    <element ref="gml:_Ring" />
  </sequence>
</complexType>
```

Encapsulates a ring to represent the surface boundary property of a surface.

#### 9.2.2.8 gml:LinearRingType, gml:LinearRing

```
<complexType name="LinearRingType">
  <complexContent>
    <extension base="gml:AbstractRingType">
      <sequence>
        <choice>
          <choice minOccurs="4" maxOccurs="unbounded">
            <element ref="gml:pos" />
            <element ref="gml:pointProperty" />
          </choice>
          <element ref="gml:posList" />
          <element ref="gml:coordinates" />
          <element ref="gml:coord" minOccurs="4" maxOccurs="unbounded">
            <annotation>
              <appinfo>deprecated</appinfo>
              <documentation>deprecated in GML version 3.0</documentation>
            </annotation>
          </element>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="LinearRing" type="gml:LinearRingType" substitutionGroup="gml:_Ring" />
```

A LinearRing is defined by four or more coordinate tuples, with linear interpolation between them; the first and last coordinates shall be coincident.

GML supports two different ways to specify the control points of a linear ring:

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this ring, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this ring (reuse of existing points).
- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this ring only. The number of direct positions in the list shall be at least four.

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

### 9.2.2.9 **gml:LinearRingPropertyType**

```
<complexType name="LinearRingPropertyType">
  <choice>
    <element ref="gml:LinearRing" />
  </choice>
</complexType>
```

Encapsulates a ring to represent properties in features or geometry collections.

### 9.2.2.10 **gml:PolygonPropertyType, gml:polygonProperty**

```
<complexType name="PolygonPropertyType">
  <sequence>
    <element ref="gml:Polygon" minOccurs="0" />
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

*This type is deprecated with GML 3 and shall not be used. It is included for backwards compatibility with GML 2. Use Surface.PropertyType instead.*

A property that has a polygon as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

```
<element name="polygonProperty" type="gml:PolygonPropertyType" />
```

*Deprecated with GML 3 and included only for backwards compatibility with GML 2. Use "surfaceProperty" instead.*

*This property element either references a polygon via the XLink-attributes or contains the polygon element.*

## 10 GML schemas – more geometric primitives

### 10.1 Overview

Beside the "simple" geometric primitives specified in the previous chapter, this clause specifies additional primitives to describe real world situations which require a more expressive geometry model.

This clause describes these primitives in the XML Schema document

- geometryPrimitives.xsd.

The GML types and elements are described in the schemas listed in Annex C. The schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:geometryPrimitives:v3.1.0

All types, elements and attributes in this XML Schema document is covered by this Clause.

## 10.2 Additional geometric primitives

### 10.2.1 1-dimensional geometric primitives

#### 10.2.1.1 gml:CurveType, gml:Curve

```
<complexType name="CurveType">
  <complexContent>
    <extension base="gml:AbstractCurveType">
      <sequence>
        <element ref="gml:segments" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="Curve" type="gml:CurveType" substitutionGroup="gml:_Curve" />
```

Curve is a 1-dimensional primitive. Curves are continuous, connected, and have a measurable length in terms of the coordinate system.

A curve is composed of one or more curve segments. Each curve segment within a curve may be defined using a different interpolation method. The curve segments are connected to one another, with the end point of each segment except the last being the start point of the next segment in the segment list.

The orientation of the curve is positive.

The element "segments" encapsulates the segments of the curve.

#### 10.2.1.2 gml:AbstractCurveSegmentType, gml:\_CurveSegment

```
<complexType name="AbstractCurveSegmentType" abstract="true">
  <attribute name="numDerivativesAtStart" type="integer" use="optional" default="0" />
  <attribute name="numDerivativesAtEnd" type="integer" use="optional" default="0" />
  <attribute name="numDerivativeInterior" type="integer" use="optional" default="0" />
</complexType>
```

Curve segment defines a homogeneous segment of a curve.

The attribute "numDerivativesAtStart" specifies the type of continuity between this curve segment and its predecessor. If this is the first curve segment in the curve, one of these values, as appropriate, is ignored. The default value of "0" means simple continuity, which is a mandatory minimum level of continuity. This level is referred to as "C 0" in mathematical texts. A value of 1 means that the function and its first derivative are continuous at the appropriate end point: "C 1" continuity. A value of "n" for any integer means the function and its first n derivatives are continuous: "C n" continuity.

The attribute "numDerivativesAtEnd" specifies the type of continuity between this curve segment and its successor. If this is the last curve segment in the curve, one of these values, as appropriate, is ignored. The default value of "0" means simple continuity, which is a mandatory minimum level of continuity. This level is referred to as "C 0" in mathematical texts. A value of 1 means that the function and its first derivative are continuous at the appropriate end point: "C 1" continuity. A value of "n" for any integer means the function and its first n derivatives are continuous: "C n" continuity.

The attribute "numDerivativesInterior" specifies the type of continuity that is guaranteed interior to the curve. The default value of "0" means simple continuity, which is a mandatory minimum level of continuity. This level is referred to as "C 0" in mathematical texts. A value of 1 means that the function and its first derivative are continuous at the appropriate end point: "C 1" continuity. A value of "n" for any integer means the function and its first n derivatives are continuous: "C n" continuity.

**NOTE** Use of these attributes is only appropriate when the basic curve definition is an underdetermined system. For example, line string segments cannot support continuity above C 0 , since there is no spare control parameter to adjust the incoming angle at the end points of the segment. Spline functions on the other hand often have extra degrees of freedom on end segments that allow them to adjust the values of the derivatives to support C 1 or higher continuity.

```
<element name="_CurveSegment" type="gml:AbstractCurveSegmentType" abstract="true" />
```

The "\_CurveSegment" element is the abstract head of the substitution group for all curve segment elements, i.e. continuous segments of the same interpolation mechanism.

#### 10.2.1.3 gml:CurveSegmentArrayType, gml:segments

```
<complexType name="CurveSegmentArrayType">
  <sequence>
    <element ref="gml:_CurveSegment" minOccurs="0" maxOccurs="unbounded" />
  </sequence>
</complexType>
```

A container for an array of curve segments.

```
<element name="segments" type="gml:CurveSegmentArrayType" />
```

This property element contains a list of curve segments. The order of the elements is significant and shall be preserved when processing the array.

#### 10.2.1.4 gml:CurveInterpolationType

```
<simpleType name="CurveInterpolationType">
  <restriction base="string">
    <enumeration value="linear" />
    <enumeration value="geodesic" />
    <enumeration value="circularArc3Points" />
    <enumeration value="circularArc2PointWithBulge" />
    <enumeration value="circularArcCenterPointWithRadius" />
    <enumeration value="elliptical" />
    <enumeration value="clothoid" />
    <enumeration value="conic" />
    <enumeration value="polynomialSpline" />
    <enumeration value="cubicSpline" />
    <enumeration value="rationalSpline" />
  </restriction>
</simpleType>
```

CurveInterpolationType is a list of codes that may be used to identify the interpolation mechanisms specified by an application schema.

#### 10.2.1.5 gml:LineStringSegmentType, gml:LineStringSegment

```
<complexType name="LineStringSegmentType">
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <choice>
          <choice minOccurs="2" maxOccurs="unbounded">
            <element ref="gml:pos" />
            <element ref="gml:pointProperty" />
          </choice>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

        <element ref="gml:posList" />
        <element ref="gml:coordinates" />
    </choice>
</sequence>
<attribute name="interpolation" type="gml:CurveInterpolationType" fixed="linear" />
</extension>
</complexContent>
</complexType>

<element name="LineStringSegment" type="gml:LineStringSegmentType"
substitutionGroup="gml:_CurveSegment" />
```

A LineStringSegment is a curve segment that is defined by two or more coordinate tuples, with linear interpolation between them.

NOTE LineStringSegment implements GM\_LineString of ISO 19107.

GML supports two different ways to specify the control points of a curve segment.

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the coordinate list shall be at least two and include the start and end point.

The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For a LineStringSegment the interpolation is fixed as "linear".

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### 10.2.1.6 gml:ArcStringType, gml:ArcString

```

<complexType name="ArcStringType">
    <complexContent>
        <extension base="gml:AbstractCurveSegmentType">
            <sequence>
                <choice>
                    <choice minOccurs="3" maxOccurs="unbounded">
                        <element ref="gml:pos" />
                        <element ref="gml:pointProperty" />
                        <element ref="gml:pointRep" />
                    </choice>
                    <element ref="gml:posList" />
                    <element ref="gml:coordinates" />
                </choice>
            </sequence>
            <attribute name="interpolation" type="gml:CurveInterpolationType" fixed="circularArc3Points" />
            <attribute name="numArc" type="integer" use="optional" />
        </extension>
    </complexContent>
</complexType>

<element name="ArcString" type="gml:ArcStringType" substitutionGroup="gml:_CurveSegment" />
```

An ArcString is a curve segment that uses three-point circular arc interpolation.

GML supports two different ways to specify the control points of a curve segment.

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the coordinate list shall be at least three and include the start and end point.

The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For an ArcString the interpolation is fixed as "circularArc3Points".

The number of arcs in the arc string can be explicitly stated in the attribute "numArc". The number of control points in the arc string shall be  $2 * \text{numArc} + 1$ .

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### 10.2.1.7 `gml:ArcType`, `gml:Arc`

```

<complexType name="ArcType">
  <complexContent>
    <restriction base="gml:ArcStringType">
      <sequence>
        <choice>
          <choice minOccurs="3" maxOccurs="3">
            <element ref="gml:pos" />
            <element ref="gml:pointProperty" />
            <element ref="gml:pointRep" />
          </choice>
          <element ref="gml:posList" />
          <element ref="gml:coordinates" />
        </choice>
      </sequence>
      <attribute name="numArc" type="integer" use="optional" fixed="1" />
    </restriction>
  </complexContent>
</complexType>

<element name="Arc" type="gml:ArcType" substitutionGroup="gml:ArcString" />

```

An Arc is an arc string with only one arc unit, i.e. three control points.

GML supports two different ways to specify the control points of a curve segment.

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).

- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the coordinate list shall be three and include the start and end point.

An arc is an arc string consisting of a single arc, the attribute "numArc" is fixed to "1".

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### 10.2.1.8 `gml:CircleType`, `gml:Circle`

```
<complexType name="CircleType">
  <complexContent>
    <extension base="gml:ArcType" />
  </complexContent>
</complexType>

<element name="Circle" type="gml:CircleType" substitutionGroup="gml:Arc" />
```

A Circle is an arc whose ends coincide to form a simple closed loop. The "start" and "end" bearing are equal and shall be the bearing for the first controlPoint listed. The three control points shall be distinct non-co-linear points for the Circle to be unambiguously defined. The arc is simply extended past the third control point until the first control point is encountered.

#### 10.2.1.9 `gml:ArcStringByBulgeType`, `gml:ArcStringByBulge`

```
<complexType name="ArcStringByBulgeType">
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <choice>
          <choice minOccurs="2" maxOccurs="unbounded">
            <element ref="gml:pos" />
            <element ref="gml:pointProperty" />
            <element ref="gml:pointRep" />
          </choice>
          <element ref="gml:posList" />
          <element ref="gml:coordinates" />
        </choice>
        <element name="bulge" type="gml:double" maxOccurs="unbounded"/>
        <element name="normal" type="gml:VectorType" maxOccurs="unbounded" />
      </sequence>
      <attribute name="interpolation" type="gml:CurveInterpolationType"
        fixed="circularArc2PointWithBulge" />
      <attribute name="numArc" type="integer" use="optional" />
    </extension>
  </complexContent>
</complexType>

<element name="ArcStringByBulge" type="gml:ArcStringByBulgeType" substitutionGroup="gml:_CurveSegment" />
```

This variant of the arc computes the mid points of the arcs instead of storing the coordinates directly. The control point sequence consists of the start and end points of each arc plus the bulge.

GML supports two different ways to specify the control points of a curve segment.

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain

a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).

- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the coordinate list shall be at least two and include the start and end point.

The element "bulge" controls the offset of each arc's midpoint. The "bulge" is the real number multiplier for the normal that determines the offset direction of the midpoint of each arc. The length of the bulge sequence is exactly 1 less than the length of the control point array, since a bulge is needed for each pair of adjacent points in the control point array. The bulge is not given by a distance, since it is simply a multiplier for the normal.

The midpoint of the resulting arc is given by:  $\text{midPoint} = ((\text{startPoint} + \text{endPoint})/2.0) + \text{bulge} * \text{normal}$ .

The element "normal" is a vector normal (perpendicular) to the chord of the arc, the line joining the first and last point of the arc. In a 2D coordinate system, there are only two possible directions for the normal, and it is often given as a signed real, indicating its length, with a positive sign indicating a left turn angle from the chord line, and a negative sign indicating a right turn from the chord. In 3D, the normal determines the plane of the arc, along with the start and endPoint of the arc.

The normal is usually a unit vector, but this is not absolutely necessary. If the normal is a zero vector, the geometric object becomes equivalent to the straight line between the two end points. The length of the normal sequence is exactly the same as for the bulge sequence, 1 less than the control point sequence length.

The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For an ArcStringByBulge the interpolation is fixed as "circularArc2PointWithBulge".

The number of arcs in the arc string can be explicitly stated in the attribute "numArc". The number of control points in the arc string shall be numArc + 1.

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### 10.2.1.10 **gml:ArcByBulgeType, gml:ArcByBulge**

```

<complexType name="ArcByBulgeType">
  <complexContent>
    <restriction base="gml:ArcStringByBulgeType">
      <sequence>
        <choice>
          <choice minOccurs="2" maxOccurs="2">
            <element ref="gml:pos" />
            <element ref="gml:pointProperty" />
            <element ref="gml:pointRep" />
          </choice>
          <element ref="gml:posList" />
          <element ref="gml:coordinates" />
        </choice>
        <element name="bulge" type="gml:double" />
        <element name="normal" type="gml:VectorType" />
      </sequence>
      <attribute name="numArc" type="integer" use="optional" fixed="1" />
    </restriction>
  </complexContent>
</complexType>

```

```
<element name="ArcByBulge" type="gml:ArcByBulgeType" substitutionGroup="gml:ArcStringByBulge" />
```

An ArcByBulge is an arc string with only one arc unit, i.e. two control points and one bulge.

GML supports two different ways to specify the control points of a curve segment.

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the coordinate list shall be two and include the start and end point.

The element "bulge" controls the offset of each arc's midpoint. The "bulge" is the real number multiplier for the normal that determines the offset direction of the midpoint of each arc. The length of the bulge sequence is exactly 1 less than the length of the control point array, since a bulge is needed for each pair of adjacent points in the control point array. The bulge is not given by a distance, since it is simply a multiplier for the normal.

The midpoint of the resulting arc is given by:  $\text{midPoint} = (\text{startPoint} + \text{endPoint})/2.0 + \text{bulge} * \text{normal}$ .

For an Arc there is exactly one bulge.

The element "normal" is a vector normal (perpendicular) to the chord of the arc, the line joining the first and last point of the arc. In a 2D coordinate system, there are only two possible directions for the normal, and it is often given as a signed real, indicating its length, with a positive sign indicating a left turn angle from the chord line, and a negative sign indicating a right turn from the chord. In 3D, the normal determines the plane of the arc, along with the start and endPoint of the arc.

The normal is usually a unit vector, but this is not absolutely necessary. If the normal is a zero vector, the geometric object becomes equivalent to the straight line between the two end points. The length of the normal sequence is exactly the same as for the bulge sequence, 1 less than the control point sequence length.

For an Arc there is exactly one normal vector.

An arc is an arc string consisting of a single arc, the attribute "numArc" is fixed to "1".

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### 10.2.1.11 gml:ArcByCenterPointType, gml:ArcByCenterPoint

```
<complexType name="ArcByCenterPointType">
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <choice>
          <choice>
            <element ref="gml:pos" />
            <element ref="gml:pointProperty" />
            <element ref="gml:pointRep" />
          </choice>
          <element ref="gml:posList" />
          <element ref="gml:coordinates" />
        </choice>
        <element name="radius" type="gml:LengthType" />
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

<element name="startAngle" type="gml:AngleType" minOccurs="0" />
<element name="endAngle" type="gml:AngleType" minOccurs="0" />
</sequence>
<attribute name="interpolation" type="gml:CurveInterpolationType"
    fixed="circularArcCenterPointWithRadius" />
<attribute name="numArc" type="integer" use="required" fixed="1" />
</extension>
</complexContent>
</complexType>

<element name="ArcByCenterPoint" type="gml:ArcByCenterPointType" substitutionGroup="gml:_CurveSegment" />
```

This variant of the arc requires that the points on the arc have to be computed instead of storing the coordinates directly. The control point is the center point of the arc plus the radius and the bearing at start and end. This representation can be used only in 2D.

GML supports two different ways to specify the control points of a curve segment.

- A "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) element. The "pos" element contains a center point that is only part of this curve segment, a "pointProperty" element contains a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
- The "posList" element can be used to specify the coordinates of the center point, too. The number of direct positions in the coordinate list shall be one.

The element "radius" specifies the radius of the arc.

The element "startAngle" specifies the bearing of the arc at the start.

The element "endAngle" specifies the bearing of the arc at the end.

The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For an ArcByCenterPoint the interpolation is fixed as "circularArcCenterPointWithRadius".

Since this type describes always a single arc, the attribute "numArc" is fixed to "1".

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### 10.2.1.12 gml:CircleByCenterPointType, gml:CircleByCenterPoint

```

<complexType name="CircleByCenterPointType">
    <complexContent>
        <extension base="gml:ArcByCenterPointType" />
    </complexContent>
</complexType>

<element name="CircleByCenterPoint" type="gml:CircleByCenterPointType"
    substitutionGroup="gml:ArcByCenterPoint" />
```

A CircleByCenterPoint is an ArcByCenterPoint with identical start and end angle to form a full circle. Again, this representation can be used only in 2D.

#### 10.2.1.13 gml:CubicSplineType, gml:CubicSpline

```
<complexType name="CubicSplineType">
```

```

<complexContent>
  <extension base="gml:AbstractCurveSegmentType">
    <sequence>
      <choice>
        <choice minOccurs="2" maxOccurs="unbounded">
          <element ref="gml:pos" />
          <element ref="gml:pointProperty" />
          <element ref="gml:pointRep" />
        </choice>
        <element ref="gml:posList" />
        <element ref="gml:coordinates" />
      </choice>
      <element name="vectorAtStart" type="gml:VectorType" />
      <element name="vectorAtEnd" type="gml:VectorType" />
    </sequence>
    <attribute name="interpolation" type="gml:CurveInterpolationType" fixed="cubicSpline" />
    <attribute name="degree" type="integer" fixed="3" />
  </extension>
</complexContent>
</complexType>

<element name="CubicSpline" type="gml:CubicSplineType" substitutionGroup="gml:_CurveSegment" />

```

Cubic splines are similar to line strings in that they are a sequence of segments each with its own defining function. A cubic spline uses the control points and a set of derivative parameters to define a piecewise 3rd degree polynomial interpolation. Unlike line-strings, the parameterization by arc length is not necessarily still a polynomial.

The function describing the curve shall be C2, that is, have a continuous 1st and 2nd derivative at all points, and pass through the controlPoints in the order given. Between the control points, the curve segment is defined by a cubic polynomial. At each control point, the polynomial changes in such a manner that the 1st and 2nd derivative vectors are the same from either side. The control parameters record shall contain vectorAtStart, and vectorAtEnd which are the unit tangent vectors at controlPoint[1] and controlPoint[n] where n = controlPoint.count.

NOTE Only the direction of the vectors is relevant, not their length.

GML supports two different ways to specify the control points of a curve segment.

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the coordinate list shall be at least three and include the start and end point.

The element "vectorAtStart" is the unit tangent vector at the start point of the spline.

The element "vectorAtEnd" is the unit tangent vector at the end point of the spline.

The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For a CubicSpline the interpolation is fixed as "cubicSpline".

The attribute "degree" shall be the degree of the polynomial used for interpolation in this spline. Therefore the degree for a cubic spline is fixed to "3".

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### 10.2.1.14 **gml:BSplineType**, **gml:BSpline**

```

<complexType name="BSplineType">
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <choice>
          <choice minOccurs="0" maxOccurs="unbounded">
            <element ref="gml:pos" />
            <element ref="gml:pointProperty" />
            <element ref="gml:pointRep" />
          </choice>
          <element ref="gml:posList" />
          <element ref="gml:coordinates" />
        </choice>
        <element name="degree" type="nonNegativeInteger" />
        <element name="knot" type="gml:Knot.PropertyType" minOccurs="2" maxOccurs="unbounded" />
      </sequence>
      <attribute name="interpolation" type="gml:CurveInterpolationType" default="polynomialSpline" />
      <attribute name="isPolynomial" type="boolean" use="optional" />
      <attribute name="knotType" type="gml:KnotTypesType" use="optional" />
    </extension>
  </complexContent>
</complexType>

<element name="BSpline" type="gml:BSplineType" substitutionGroup="gml:_CurveSegment" />

```

A B-Spline is a piecewise parametric polynomial or rational curve described in terms of control points and basis functions. Knots are breakpoints on the curve that connect its pieces.

They are given as a non-decreasing sequence of real numbers. If the weights in the knots are equal then it is a polynomial spline. The degree is the algebraic degree of the basis functions.

GML supports two different ways to specify the control points of a curve segment.

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only.

The property element "degree" shall be the degree of the polynomial used for interpolation in this spline.

The property element "knot" shall be the sequence of distinct knots used to define the spline basis functions.

The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For a BSpline the interpolation can be either "polynomialSpline" or "rationalSpline", default is "polynomialSpline".

The attribute "isPolynomial" is set to "true" if this is a polynomial spline.

The attribute "knotType" gives the type of knot distribution used in defining this spline. This is for information only and is set according to the different construction-functions.

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.

#### 10.2.1.15 **gml:KnotType**, **gml:Knot.PropertyType**

```
<complexType name="KnotType">
  <sequence>
    <element name="value" type="double" />
    <element name="multiplicity" type="nonNegativeInteger" />
    <element name="weight" type="double" />
  </sequence>
</complexType>
```

A knot is a breakpoint on a piecewise spline curve.

The property element "value" is the value of the parameter at the knot of the spline. The sequence of knots shall be a non-decreasing sequence. That is, each knot's value in the sequence shall be equal to or greater than the previous knot's value. The use of equal consecutive knots is normally handled using the multiplicity.

The property element "multiplicity" is the multiplicity of this knot used in the definition of the spline (with the same weight).

The property element "weight" is the value of the averaging weight used for this knot of the spline.

```
<complexType name="Knot.PropertyType">
  <sequence>
    <element name="Knot" type="gml:KnotType" />
  </sequence>
</complexType>
```

Encapsulates a knot to use it in a geometric type.

#### 10.2.1.16 **gml:KnotTypesType**

```
<simpleType name="KnotTypesType">
  <restriction base="string">
    <enumeration value="uniform" />
    <enumeration value="quasiUniform" />
    <enumeration value="piecewiseBezier" />
  </restriction>
</simpleType>
```

Defines allowed values for the knots' type. Uniform knots implies that all knots are of multiplicity 1 and they differ by a positive constant from the preceding knot. Knots are quasi-uniform if they are of multiplicity (degree + 1) at the ends, of multiplicity 1 elsewhere, and they differ by a positive constant from the preceding knot.

#### 10.2.1.17 **gml:BezierType**, **gml:Bezier**

```
<complexType name="BezierType">
  <complexContent>
    <restriction base="gml:BSplineType">
      <sequence>
        <choice>
          <choice minOccurs="0" maxOccurs="unbounded">
            <element ref="gml:pos" />
            <element ref="gml:pointProperty" />
            <element ref="gml:pointRep" />
          </choice>
          <element ref="gml:posList" />
        </choice>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
```

```

<element ref="gml:coordinates" />
</choice>
<element name="degree" type="nonNegativeInteger" />
<element name="knot" type="gml:Knot.PropertyType" minOccurs="2" maxOccurs="2" />
</sequence>
<attribute name="interpolation" type="gml:CurveInterpolationType" fixed="polynomialSpline" />
<attribute name="isPolynomial" type="boolean" fixed="true" />
<attribute name="knotType" type="gml:KnotTypesType" use="prohibited" />
</restriction>
</complexContent>
</complexType>

<element name="Bezier" type="gml:BezierType" substitutionGroup="gml:BSpline" />

```

Bezier curves are polynomial splines that use Bezier or Bernstein polynomials for interpolation purposes. It is a special case of the B-Spline curve with two knots.

GML supports two different ways to specify the control points of a curve segment.

- A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
- The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only.

The property element "degree" shall be the degree of the polynomial used for interpolation in this spline.

The property element "knot" shall be the sequence of distinct knots used to define the spline basis functions.

The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For a Bezier the interpolation is fixed as "polynomialSpline".

The attribute "isPolynomial" shall be "true" as this is a polynomial spline.

The property "knotType" is not relevant for Bezier curve segments.

*The use of the element "coordinates" is deprecated with GML version 3.1.0. Use "posList" instead.*

*The use of the elements "coord" is deprecated. Use "pos" instead, the element is included only for backwards compatibility with previous versions of GML.*

#### 10.2.1.18 gml:OrientableCurveType, gml:OrientableCurve, gml:baseCurve

```

<complexType name="OrientableCurveType">
<complexContent>
<extension base="gml:AbstractCurveType">
<sequence>
<element ref="gml:baseCurve" />
</sequence>
<attribute name="orientation" type="gml:SignType" default="+" />
</extension>
</complexContent>
</complexType>

<element name="baseCurve" type="gml:Curve.PropertyType" />
<element name="OrientableCurve" type="gml:OrientableCurveType" substitutionGroup="gml:_Curve" />

```

OrientableCurve consists of a curve and an orientation. If the orientation is "+", then the OrientableCurve is identical to the baseCurve. If the orientation is "-", then the OrientableCurve is related to another \_Curve with a parameterization that reverses the sense of the curve traversal.

The element "baseCurve" references or contains the base curve. The property "baseCurve" either references the base curve via the XLink-attributes or contains the curve element. A curve element is any element which is substitutable for "\_Curve". The base curve has positive orientation.

NOTE This definition allows for a nested structure, i.e. an OrientableCurve may use another OrientableCurve as its base curve.

If the attribute "orientation" is "+", then the OrientableCurve is identical to the baseCurve. If the orientation is "-", then the OrientableCurve is related to another \_Curve with a parameterization that reverses the sense of the curve traversal. "+" is the default value.

#### 10.2.1.19 **gml:OffsetCurveType**, **gml:OffsetCurve**

```
<complexType name="OffsetCurveType">
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <element name="offsetBase" type="gml:CurvePropertyType"/>
        <element name="distance" type="gml:LengthType"/>
        <element name="refDirection" type="gml:VectorType" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="OffsetCurve" type="gml:OffsetCurveType" substitutionGroup="gml:_CurveSegment"/>
```

An offset curve is a curve at a constant distance from the basis curve. Offsets can be useful as cheap and simple alternatives to constructing curves that are offsets by definition.

OffsetBase is a reference to the curve from which this curve is defined as an offset.

Distance is the distance at which the offset curve is generated from the basis curve. In 2D systems, positive distances are to be to the left of the basis curve, and the negative distances are to be to the right of the basis curve.

refDistance is used to define the vector direction of the offset curve from the basis curve. It can be omitted in the 2D case, where the distance can be positive or negative. In that case, distance defines left side (positive distance) or right side (negative distance) with respect to the tangent to the basis curve.

In 3D, the basis curve shall have a well defined tangent direction for every point. The offset curve at any point in 3D, the basis curve shall have a well-defined tangent direction for every point. The offset curve at any point (parameter) on the basis curve "c" is in the direction

$$\vec{s} = \vec{v} \times \vec{t} \text{ where } \vec{v} = c.refDirection() \text{ and } \vec{t} = c.tangent()$$

For the offset direction to be well-defined, v shall not on any point of the curve be in the same, or opposite, direction as t.

The default value of the refDirection shall be the local co-ordinate axis vector for elevation, which indicates up for the curve in a geographic sense.

NOTE If the refDirection is the positive tangent to the local elevation axis ("points upward"), then the offset vector points to the left of the curve when viewed from above.

### 10.2.1.20 gml:AffinePlacementType, gml:AffinePlacement

```

<complexType name="AffinePlacementType">
  <sequence>
    <element name="location" type="gml:DirectPositionType"/>
    <element name="refDirection" type="gml:VectorType" maxOccurs="unbounded"/>
    <element name="inDimension" type="positiveInteger"/>
    <element name="outDimension" type="positiveInteger"/>
  </sequence>
</complexType>

<element name="AffinePlacement" type="AffinePlacementType"/>

```

A placement takes a standard geometric construction and places it in geographic space. It defines a transformation from a constructive parameter space to the co-ordinate space of the co-ordinate reference system being used. Parameter spaces in formulae in this International Standard are given as  $(u, v)$  in 2D and  $(u, v, w)$  in 3D. Co-ordinate reference systems positions are given in formulae, in this International Standard, by either  $(x, y)$  in 2D, or  $(x, y, z)$  in 3D.

Affine placements are defined by linear transformations from parameter space to the target co-ordinate space. 2-dimensional Cartesian parameter space,  $(u, v)$  transforms into 3-dimensional co-ordinate reference systems  $, (x, y, z)$  by using an affine transformation,  $(u, v) \rightarrow (x, y, z)$  which is defined:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} u_x & v_x \\ u_y & v_y \\ u_z & v_z \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix}$$

Then, given this equation, the location element of the AffinePlacement is the direct position  $(x_0, y_0, z_0)$ , which is the target position of the origin in  $(u, v)$ . The two reference directions  $(u_x, u_y, u_z)$  and  $(v_x, v_y, v_z)$  are the target directions of the unit vectors at the origin in  $(u, v)$ .

The “location” property gives the target of the parameter space origin. This is the vector  $(x_0, y_0, z_0)$  in the formulae above.

The attribute “refDirection” gives the target directions for the co-ordinate basis vectors of the parameter space. These are the columns of the matrix in the formulae given above. The number of directions given shall be inDimension. The dimension of the directions shall be outDimension.

### 10.2.1.21 gml:ClothoidType, gml:Clothoid

```

<complexType name="ClothoidType">
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <element name="refLocation">
          <complexType>
            <sequence>
              <element ref="gml:AffinePlacement"/>
            </sequence>
          </complexType>
        </element>
        <element name="scaleFactor" type="decimal"/>
        <element name="startParameter" type="double"/>
        <element name="endParameter" type="double"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="Clothoid" type="gml:ClothoidType" substitutionGroup="gml:_CurveSegment"/>

```

A clothoid, or Cornu's spiral, is plane curve whose curvature is a fixed function of its length. In suitably chosen co-ordinates it is given by Fresnel's integrals:

$$x(t) = \int_0^t \cos\left(\frac{A\tau^2}{2}\right) d\tau \quad \text{and} \quad y(t) = \int_0^t \sin\left(\frac{A\tau^2}{2}\right) d\tau$$

This geometry is mainly used as a transition curve between curves of type straight line to circular arc or circular arc to circular arc. With this curve type it is possible to achieve a C2-continuous transition between the above mentioned curve types. One formula for the clothoid is  $A^*A = R*t$  where A is a constant, R is the varying radius of curvature along the curve and t is the length along the curve and given in the Fresnel integrals.

The "refLocation" is an affine mapping that places the curve defined by the Fresnel Integrals into the co-ordinate reference system of this object.

The startParameter is the arc length distance from the inflection point that will be the start point for this curve segment. This shall be lower limit used in the Fresnel integral and is the value of the constructive parameter of this curve segment at its start point. The startParameter can either be positive or negative.

**NOTE** If 0.0 (zero), lies between the startParameter and the endParameter of the clothoid, then the curve goes through the clothoid's inflection point, and the direction of its radius of curvature, given by the second derivative vector, changes sides with respect to the tangent vector. The term "length" distance for the parameter "t" is applicable only in the parameter space and its relation to arc length after use of the placement, and with respect to the co-ordinate reference system of the curve is not deterministic.

The endParameter is the arc length distance from the inflection point that will be the end point for this curve segment. This shall be upper limit used in the Fresnel integral and is the value of the constructive parameter of this curve segment at its start point. The startParameter can either be positive or negative.

#### 10.2.1.22 gml:GeodesicStringType, gml:GeodesicString

```

<complexType name="GeodesicStringType">
    <complexContent>
        <extension base="gml:AbstractCurveSegmentType">
            <choice>
                <element ref="gml:posList"/>
                <group ref="gml:geometricPositionGroup" minOccurs="2" maxOccurs="unbounded"/>
            </choice>
            <attribute name="interpolation" type="gml:CurveInterpolationType" fixed="geodesic">
            </attribute>
        </extension>
    </complexContent>
</complexType>

<element name="GeodesicString" type="gml:GeodesicStringType" substitutionGroup="gml:_CurveSegment"/>
```

A GeodesicString consists of sequence of geodesic segments. The type essentially combines a sequence of Geodesic into a single object. The GeodesicString is computed from two or more positions and an interpolation using geodesics defined from the geoid (or ellipsoid) of the co-ordinate reference system being used.

GML supports two different ways to specify the control points of a curve segment.

A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may

be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).

The “posList” element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only.

The number of direct positions in the list shall be at least two.

The attribute “interpolation” specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For an GeodesicString the interpolation is fixed as “geodesic”.

#### 10.2.1.23 **gml:GeodesicType, gml:Geodesic**

```
<complexType name="GeodesicType">
  <complexContent>
    <extension base="gml:GeodesicStringType"/>
  </complexContent>
</complexType>

<element name="Geodesic" type="gml:GeodesicType" substitutionGroup="gml:GeodesicString"/>
```

A Geodesic consists of two distinct positions joined by a geodesic curve. The control points of a Geodesic shall lie on the geodesic between its start point and end points. Between these two points, a geodesic curve defined from ellipsoid or geoid model used by the co-ordinate reference systems may be used to interpolate other positions. Any other point in the controlPoint array shall fall on this geodesic.

### 10.2.2 2-dimensional geometric primitives

#### 10.2.2.1 **gml:SurfaceType, gml:Surface**

```
<complexType name="SurfaceType">
  <complexContent>
    <extension base="gml:AbstractSurfaceType">
      <sequence>
        <element ref="gml:patches" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="Surface" type="gml:SurfaceType" substitutionGroup="gml:_Surface" />
```

A Surface is a 2-dimensional primitive and is composed of one or more surface patches. The surface patches are connected to one another.

The orientation of the surface is positive (“up”). The orientation of a surface chooses an “up” direction through the choice of the upward normal, which, if the surface is not a cycle, is the side of the surface from which the exterior boundary appears counterclockwise. Reversal of the surface orientation reverses the curve orientation of each boundary component, and interchanges the conceptual “up” and “down” direction of the surface. If the surface is the boundary of a solid, the “up” direction is usually outward. For closed surfaces, which have no boundary, the up direction is that of the surface patches, which shall be consistent with one another. Its included surface patches describe the interior structure of the Surface.

The element “patches” encapsulates the patches of the surface.

#### 10.2.2.2 **gml:AbstractSurfacePatchType, gml :\_SurfacePatch**

```
<complexType name="AbstractSurfacePatchType" abstract="true" />
```

A surface patch defines a homogenous portion of a surface.

```
<element name="_SurfacePatch" type="gml:AbstractSurfacePatchType" abstract="true" />
```

The "\_SurfacePatch" element is the abstract head of the substitution group for all surface patch elements describing a continuous portion of a surface.

#### 10.2.2.3 gml:SurfacePatchArrayPropertyType, gml:patches

```
<complexType name="SurfacePatchArrayPropertyType">
  <sequence>
    <element ref="gml:_SurfacePatch" minOccurs="0" maxOccurs="unbounded" />
  </sequence>
</complexType>
```

A container for an array of surface patches.

```
<element name="patches" type="gml:SurfacePatchArrayPropertyType" />
```

This property element contains a list of surface patches. The order of the elements is significant and shall be preserved when processing the array.

#### 10.2.2.4 gml:SurfaceInterpolationType

```
<simpleType name="SurfaceInterpolationType">
  <restriction base="string">
    <enumeration value="none" />
    <enumeration value="planar" />
    <enumeration value="spherical" />
    <enumeration value="elliptical" />
    <enumeration value="conic" />
    <enumeration value="tin" />
    <enumeration value="parametricCurve" />
    <enumeration value="polynomialSpline" />
    <enumeration value="rationalSpline" />
    <enumeration value="triangulatedSpline" />
  </restriction>
</simpleType>
```

SurfaceInterpolationType is a list of codes that may be used to identify the interpolation mechanisms specified by an application schema.

#### 10.2.2.5 gml:PolygonPatchType

```
<complexType name="PolygonPatchType">
  <complexContent>
    <extension base="gml:AbstractSurfacePatchType">
      <sequence>
        <element ref="gml:exterior" minOccurs="0" />
        <element ref="gml:interior" minOccurs="0" maxOccurs="unbounded" />
      </sequence>
      <attribute name="interpolation" type="gml:SurfaceInterpolationType" fixed="planar" />
    </extension>
  </complexContent>
</complexType>

<element name="PolygonPatch" type="gml:PolygonPatchType" substitutionGroup="gml:_SurfacePatch" />
```

A PolygonPatch is a surface patch that is defined by a set of boundary curves and an underlying surface to which these curves adhere. The curves are coplanar and the polygon uses planar interpolation in its interior. Implements GM\_Polygon of ISO 19107.

The attribute "interpolation" specifies the interpolation mechanism used for this surface patch. Currently only planar surface patches are defined in GML 3, the attribute is fixed to "planar", i.e. the interpolation method shall return points on a single plane. The boundary of the patch shall be contained within that plane.

#### 10.2.2.6 **gml:TriangleType, gml:Triangle**

```
<complexType name="TriangleType">
  <complexContent>
    <extension base="gml:AbstractSurfacePatchType">
      <sequence>
        <element ref="gml:exterior" />
      </sequence>
      <attribute name="interpolation" type="gml:SurfaceInterpolationType" fixed="planar" />
    </extension>
  </complexContent>
</complexType>

<element name="Triangle" type="gml:TriangleType" substitutionGroup="gml:_SurfacePatch" />
```

Represents a triangle as a surface with an outer boundary consisting of a linear ring. Note that this is a polygon (subtype) with no inner boundaries. The number of points in the linear ring shall be four.

The Ring (element "exterior") shall be a LinearRing and shall form a triangle, the first and the last position shall be co-incident.

The attribute "interpolation" specifies the interpolation mechanism used for this surface patch. Currently only planar surface patches are defined in GML 3, the attribute is fixed to "planar", i.e. the interpolation method shall return points on a single plane. The boundary of the patch shall be contained within that plane.

#### 10.2.2.7 **gml:RectangleType, gml:Rectangle**

```
<complexType name="RectangleType">
  <complexContent>
    <extension base="gml:AbstractSurfacePatchType">
      <sequence>
        <element ref="gml:exterior" />
      </sequence>
      <attribute name="interpolation" type="gml:SurfaceInterpolationType" fixed="planar" />
    </extension>
  </complexContent>
</complexType>

<element name="Rectangle" type="gml:RectangleType" substitutionGroup="gml:_SurfacePatch" />
```

Represents a rectangle as a surface with an outer boundary consisting of a linear ring. Note that this is a polygon (subtype) with no inner boundaries. The number of points in the linear ring shall be five.

The Ring (element "exterior") shall be a LinearRing and shall form a rectangle; the first and the last position shall be co-incident.

The attribute "interpolation" specifies the interpolation mechanism used for this surface patch. Currently only planar surface patches are defined in GML 3, the attribute is fixed to "planar", i.e. the interpolation method shall return points on a single plane. The boundary of the patch shall be contained within that plane.

#### 10.2.2.8 **gml:RingType**

```
<complexType name="RingType">
  <complexContent>
    <extension base="gml:AbstractRingType">
      <sequence>
        <element ref="gml:curveMember" maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

</sequence>
</extension>
</complexContent>
</complexType>

<element name="Ring" type="gml:RingType" substitutionGroup="gml:_Ring" />

<element name="curveMember" type="gml:CurvePropertyType" />

```

A Ring is used to represent a single connected component of a surface boundary. It consists of a sequence of curves connected in a cycle (an object whose boundary is empty).

A Ring is structurally similar to a composite curve in that the endPoint of each curve in the sequence is the startPoint of the next curve in the Sequence. Since the sequence is circular, there is no exception to this rule. Each ring, like all boundaries, is a cycle and each ring is simple.

**NOTE** Even though each Ring is simple, the boundary need not be simple. The easiest case of this is where one of the interior rings of a surface is tangent to its exterior ring.

The element "curveMember" references or contains one curve in the composite curve. The curves are contiguous, the collection of curves is ordered.

**NOTE** This definition allows for a nested structure, i.e. a CompositeCurve may use, for example, another CompositeCurve as a curve member.

This property element either references a curve via the XLink-attributes or contains the curve element. A curve element is any element which is substitutable for "gml:\_Curve".

The element is used, for example, in a Ring.

#### 10.2.2.9 gml:RingPropertyType

```

<complexType name="RingPropertyType">
  <sequence>
    <element ref="gml:Ring" />
  </sequence>
</complexType>

```

Encapsulates a ring to represent properties in features or geometry collections.

#### 10.2.2.10 gml:OrientableSurfaceType, gml:OrientableSurface, gml:baseSurface

```

<complexType name="OrientableSurfaceType">
  <complexContent>
    <extension base="gml:AbstractSurfaceType">
      <sequence>
        <element ref="gml:baseSurface" />
      </sequence>
      <attribute name="orientation" type="gml:SignType" default="+" />
    </extension>
  </complexContent>
</complexType>

<element name="baseSurface" type="gml:SurfacePropertyType" />

<element name="OrientableSurface" type="gml:OrientableSurfaceType" substitutionGroup="gml:_Surface" />

```

OrientableSurface consists of a surface and an orientation. If the orientation is "+", then the OrientableSurface is identical to the baseSurface. If the orientation is "-", then the OrientableSurface is a reference to a Surface with an up-normal that reverses the direction for this OrientableSurface, the sense of "the top of the surface".

The element “baseSurface” references or contains the base surface. The property “baseSurface” either references the base surface via the XLink-attributes or contains the surface element. A surface element is any element which is substitutable for “gml:\_Surface”. The base surface has positive orientation.

**NOTE** This definition allows for a nested structure, i.e. an OrientableSurface may use another OrientableSurface as its base surface.

If the attribute “orientation” is “+”, then the OrientableSurface is identical to the baseSurface. If the orientation is “-”, then the OrientableSurface is a reference to a Surface with an up-normal that reverses the direction for this OrientableSurface, the sense of “the top of the surface”. “+” is the default value.

#### 10.2.2.11 gml:PointGrid

```
<group name="PointGrid">
  <sequence>
    <element name="row" maxOccurs="unbounded">
      <complexType>
        <sequence>
          <group ref="gml:geometricPositionListGroup"/>
        </sequence>
      </complexType>
    </element>
  </sequence>
</group>
```

Reference points which are organised into sequences or grids (sequences of equal length sequences).

#### 10.2.2.12 gml:AbstractParametricCurveSurfaceType, gml:\_ParametricCurveSurface

```
<complexType name="AbstractParametricCurveSurfaceType">
  <complexContent>
    <extension base="gml:AbstractSurfacePatchType"/>
  </complexContent>
</complexType>

<element name="_ParametricCurveSurface" type="gml:AbstractParametricCurveSurfaceType" abstract="true"
substitutionGroup="gml:_SurfacePatch"/>
```

The surface patches that make up the parametric curve surface are all continuous families of curves, given by a constructive function of the form:

```
surface(s, t) : [a, b] × [c, d] → DirectPosition
```

By fixing the value of either parameter, we have a one-parameter family of curves.

```
ct(s) = cs(t) = surface(s, t)
```

Various possible pairs of types of parametric curve surfaces:

Surface type	Horizontal curve type	Vertical curve type
GM_Cylinder	Circle, constant radii	Line Segment
GM_Cone	Circle, decreasing radii	Line Segment
GM_Sphere	Circle of constant latitude	Circle of constant longitude
GM_BilinearGrid	Line string	Line string
GM_BicubicGrid	Cubic spline	Cubic spline

The two partial derivatives of the surface parameterisation, i and j are given by:

$$i \equiv \frac{dsurface}{ds} = \frac{d}{ds} c_i(s) = \frac{\partial}{\partial s} surface(s, t)$$

and

$$j \equiv \frac{dsurface}{dt} = \frac{d}{dt} c_i(t) = \frac{\partial}{\partial t} surface(s, t)$$

The default upNormal for the surface shall be the vector cross product of these two curve derivatives when they are both non-zero.

$$k = i \times j$$

If the co-ordinate reference system is 2D, then the vector k extends the local co-ordinate system by supplying an "upward" elevation vector. In this case the vector basis (i,j) shall be a right hand system, that is to say, the oriented angle from i to j shall be less than 180 degrees. This gives a right-handed "moving frame" of local co-ordinate axes given by  $\langle i, j \rangle$ . A moving frame is defined to be a continuous function from the geometric object to a basis for the local tangent space of that object. For curves, this is the derivative of the curve, the local tangent. For surfaces, this is a local pair of tangents. Parameterised curve surfaces have a natural moving frame and it shall be used as defined in this paragraph to define the upNormal of the surface.

**NOTE** The existence of a viable moving frame is the definition of "orientable" manifold. This is why the existence of a continuous upNormal implies that the surface is orientable. Non-orientable surfaces, such as the Möbius band and Klein bottle are counter-intuitive. These are forbidden for use in application schemas conforming to this international standard. Klein bottles cannot even be constructed in 3D space, but require 4D space for non-singular representations.

#### 10.2.2.13 **gml:AbstractGriddedSurfaceType**, **gml:\_GriddedSurface**

```

<complexType name="AbstractGriddedSurfaceType">
  <complexContent>
    <extension base="gml:AbstractParametricCurveSurfaceType">
      <sequence>
        <element name="rows" type="integer" minOccurs="0"/>
        <element name="columns" type="integer" minOccurs="0"/>
        <group ref="gml:PointGrid"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="_GriddedSurface" type="gml:AbstractGriddedSurfaceType" abstract="true"
substitutionGroup="gml:_ParametricCurveSurface"/>
```

A gridded surface is a parametric curve surface derived from a rectangular grid in the parameter space. The rows from this grid are control points for horizontal surface curves; the columns are control points for vertical surface curves. The working assumption is that for a pair of parametric co-ordinates (s, t) that the horizontal curves for each integer offset are calculated and evaluated at "s". This defines a sequence of control points:

```
<cn(s) : s = 1 .. columns>
```

From this sequence a vertical curve is calculated for "s", and evaluated at "t". In most cases, the order of calculation (horizontal-vertical vs. vertical-horizontal) does not make a difference. Where it does, the horizontal- vertical order shall be the one used.

$$\text{surface}(s, t) = \sum_{i=0}^{\text{row columns}} \sum_{j=0}^{w^2} w^2_i(s) w^2_j(t) \vec{P}_{i,j} \quad \text{where } \vec{P}_{i,j} \text{ is the control point in the } i^{\text{th}} \text{ row and } j^{\text{th}} \text{ column.}$$

Logically, any pair of curve interpolation types can lead to a subtype of GriddedSurface. The following clauses define some most commonly encountered surfaces that can be represented in this manner.

This is the double indexed sequence of control points, given in row major form.

**NOTE** There is no assumption made about the shape of the grid. For example, the positions need not effect a "21/2D" surface, consecutive points may be equal in any or all of the ordinates. Further, the curves in either or both directions may close.

The attribute "rows" gives the number of rows in the parameter grid.

The attribute "columns" gives the number of columns in the parameter grid.

#### 10.2.2.14 gml:ConeType, gml:Cone

```
<complexType name="ConeType">
  <complexContent>
    <extension base="gml:AbstractGriddedSurfaceType">
      <attribute name="horizontalCurveType" type="gml:CurveInterpolationType"
        fixed="circularArc3Points"/>
      <attribute name="verticalCurveType" type="gml:CurveInterpolationType" fixed="linear"/>
    </extension>
  </complexContent>
</complexType>

<element name="Cone" type="gml:ConeType" substitutionGroup="gml:_GriddedSurface"/>
```

A cone is a gridded surface given as a family of conic sections whose control points vary linearly.

**NOTE!** A 5-point ellipse with all defining positions identical is a point. Thus, a truncated elliptical cone can be given as a 2x5 set of control points <<P1, P1, P1, P1, P1>, <P2, P3, P4, P5, P6>>. P1 is the apex of the cone. P2, P3, P4, P5 and P6 are any five distinct points around the base ellipse of the cone. If the horizontal curves are circles as opposed to ellipses, the circular cone can be constructed using <<P1, P1, P1>, <P2, P3, P4>>.

#### 10.2.2.15 gml:CylinderType, gmlCylinder

```
<complexType name="CylinderType">
  <complexContent>
    <extension base="gml:AbstractGriddedSurfaceType">
      <attribute name="horizontalCurveType" type="gml:CurveInterpolationType"
        fixed="circularArc3Points"/>
    </extension>
  </complexContent>
</complexType>
```

```

        <attribute name="verticalCurveType" type="gml:CurveInterpolationType" fixed="linear"/>
    </extension>
</complexContent>
</complexType>

<element name="Cylinder" type="gml:CylinderType" substitutionGroup="gml:_GriddedSurface"/>

```

A cylinder is a gridded surface given as a family of circles whose positions vary along a set of parallel lines, keeping the cross sectional horizontal curves of a constant shape.

NOTE! Given the same working assumptions as in the previous note, a Cylinder can be given by two circles, giving us the control points of the form <<P1, P2, P3>, <P4, P5, P6>>.

#### 10.2.2.16 gml:SphereType, gml:Sphere

```

<complexType name="SphereType">
    <complexContent>
        <extension base="gml:AbstractGriddedSurfaceType">
            <attribute name="horizontalCurveType" type="gml:CurveInterpolationType"
fixed="circularArc3Points"/>
            <attribute name="verticalCurveType" type="gml:CurveInterpolationType" fixed="circularArc3Points"/>
        </extension>
    </complexContent>
</complexType>

<element name="Sphere" type="gml:SphereType" substitutionGroup="gml:_GriddedSurface"/>

```

A sphere is a gridded surface given as a family of circles whose positions vary linearly along the axis of the sphere, and whose radius varies in proportion to the cosine function of the central angle. The horizontal circles resemble lines of constant latitude, and the vertical arcs resemble lines of constant longitude.

NOTE! If the control points are sorted in terms of increasing longitude, and increasing latitude, the upNormal of a sphere is the outward normal.

EXAMPLE If we take a gridded set of latitudes and longitudes in degrees,(u,v) such as

(-90, -180)	(-90, -90)	(-90,0)	(-90, 90)	(-90, 180)
(-45, -180)	(-45, -90)	(-45,0)	(-45, 90)	(-45, 180)
(0, -180)	(0, -90)	(0,0)	(0, 90)	(0, 180)
(45, -180)	(45, -90)	(45,0)	(45, 90)	(45, 180)
(90, -180)	(90, -90)	(90,0)	(90, 90)	(90, 180)

And map these points to 3D using the usual equations (where R is the radius of the required sphere).

$$z = R \sin u$$

$$x = (R \cos u)(\sin v)$$

$$y = (R \cos u)(\cos v)$$

We have a sphere of Radius R, centred at (0,0), as a gridded surface. Notice that the entire first row and the entire last row of the control points map to a single point in each 3D Euclidean space, North and South poles respectively, and that each horizontal curve closes back on itself forming a geometric cycle. This gives us a metrically bounded (of finite size), topologically unbounded (not having a boundary, a cycle) surface.

### 10.2.2.17 **gml:PolyhedralSurfaceType, gml:PolyhedralSurface**

```

<complexType name="PolyhedralSurfaceType">
    <complexContent>
        <restriction base="gml:SurfaceType">
            <sequence>
                <group ref="gml:StandardObjectProperties"/>
                <element ref="gml:polygonPatches"/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>

<element name="PolyhedralSurface" type="gml:PolyhedralSurfaceType" substitutionGroup="gml:Surface"/>

```

A polyhedral surface is a surface composed of polygon surfaces connected along their common boundary curves. This differs from the surface type only in the restriction on the types of surface patches acceptable.

This property encapsulates the patches of the polyhedral surface.

### 10.2.2.18 **gml:PolygonPatchArrayPropertyType, gml:PolygonPatches**

```

<complexType name="PolygonPatchArrayPropertyType">
    <complexContent>
        <restriction base="gml:SurfacePatchArrayPropertyType">
            <sequence>
                <element ref="gml:PolygonPatch" minOccurs="0" maxOccurs="unbounded"/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>

<element name="polygonPatches" type="gml:PolygonPatchArrayPropertyType" substitutionGroup="gml:patches"/>

```

This type defines a container for an array of polygon patches.

The property element contains a list of polygon patches. The order of the patches is significant and shall be preserved when processing the list.

### 10.2.2.19 **gml:TriangulatedSurfaceType, gml:TriangulatedSurface**

```

<complexType name="TrianglePatchArrayPropertyType">
    <complexContent>
        <restriction base="gml:SurfacePatchArrayPropertyType">
            <sequence>
                <element ref="gml:Triangle" minOccurs="0" maxOccurs="unbounded"/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>

<element name="trianglePatches" type="TrianglePatchArrayPropertyType" substitutionGroup="gml:patches"/>

```

A triangulated surface is a polyhedral surface that is composed only of triangles. There is no restriction on how the triangulation is derived.

The triangle patches property encapsulates the patches of the triangulated surface.

#### 10.2.2.20 gml:TrianglePatchArrayPropertyType, gml:TrianglePatches

```
<complexType name="TriangulatedSurfaceType">
  <complexContent>
    <restriction base="gml:SurfaceType">
      <sequence>
        <group ref="gml:StandardObjectProperties"/>
        <element ref="gml:trianglePatches"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>

<element name="TriangulatedSurface" type="gml:TriangulatedSurfaceType" substitutionGroup="gml:Surface"/>
```

This type defines a container for an array of triangle patches.

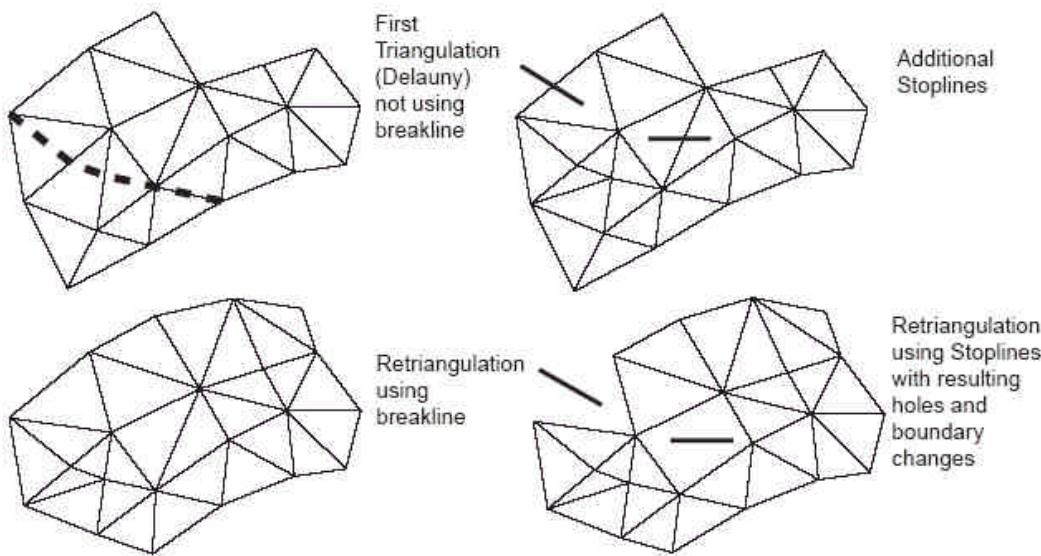
This property element contains a list of triangle patches. The order of the patches is significant and shall be preserved when processing the list.

#### 10.2.2.21 gml:TinType, gml:Tin

```
<complexType name="TinType">
  <complexContent>
    <extension base="gml:TriangulatedSurfaceType">
      <sequence>
        <element name="stopLines" type="gml:LineStringSegmentArrayPropertyType"
          minOccurs="0" maxOccurs="unbounded"/>
        <element name="breakLines" type="gml:LineStringSegmentArrayPropertyType"
          minOccurs="0" maxOccurs="unbounded"/>
        <element name="maxLength" type="gml:LengthType"/>
        <element name="controlPoint">
          <complexType>
            <choice>
              <element ref="gml:posList"/>
              <group ref="gml:geometricPositionGroup"
                minOccurs="3" maxOccurs="unbounded"/>
            </choice>
          </complexType>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="Tin" type="gml:TinType" substitutionGroup="gml:TriangulatedSurface"/>
```

A tin is a triangulated surface that uses the Delaunay algorithm or a similar algorithm complemented with consideration of breaklines, stoplines, and maximum length of triangle sides. These networks satisfy the Delaunay's criterion away from the modifications: For each triangle in the network, the circle passing through its vertices does not contain, in its interior, the vertex of any other triangle.



Stoplines are lines where the local continuity or regularity of the surface is questionable. In the area of these pathologies, triangles intersecting a stopline shall be removed from the tin surface, leaving holes in the surface. If coincidence occurs on surface boundary triangles, the result shall be a change of the surface boundary. Stoplines contains all these pathological segments as a set of line strings.

Breaklines are lines of a critical nature to the shape of the surface, representing local ridges, or depressions (such as drainage lines) in the surface. As such their constituent segments shall be included in the tin even if doing so violates the Delaunay criterion. Break lines contains these critical segments as a set of line strings.

Areas of the surface where data is not sufficiently dense to assure reasonable calculation shall be removed by adding a retention criterion for triangles based on the length of their sides. For many triangle sides exceeding maximum length, the adjacent triangles to that triangle side shall be removed from the surface.

The corners of the triangles in the TIN are often referred to as pots. ControlPoint shall contain a set of the positions used as posts for this TIN. Since each TIN contains triangles, there shall be at least 3 posts. The order in which these points are given does not affect the surface that is represented. Application schemas may add information based on ordering of control points to facilitate the reconstruction of the TIN from the control points.

#### 10.2.2.22 gml:LineStringSegmentArrayType

```
<complexType name="LineStringSegmentArrayType">
  <sequence>
    <element ref="gml:LineStringSegment" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
```

### 10.2.3 3-dimensional geometric primitives

#### 10.2.3.1 gml:AbstractSolidType, gml:\_Solid

```
<complexType name="AbstractSolidType">
  <complexContent>
    <extension base="gml:AbstractGeometricPrimitiveType" />
  </complexContent>
</complexType>
```

An abstraction of a solid to support the different levels of complexity. A solid is always contiguous.

```
<element name="_Solid" type="gml:AbstractSolidType" abstract="true"  
substitutionGroup="gml:_GeometricPrimitive" />
```

The "\_Solid" element is the abstract head of the substitution group for all (continuous) solid elements.

#### 10.2.3.2 gml:SolidPropertyType, gml:solidProperty

```
<complexType name="SolidPropertyType">  
  <sequence>  
    <element ref="gml:_Solid" minOccurs="0" />  
  </sequence>  
  <attributeGroup ref="gml:AssociationAttributeGroup" />  
</complexType>
```

A property that has a solid as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

```
<element name="solidProperty" type="gml:SolidPropertyType" />
```

This property element either references a solid via the XLink-attributes or contains the solid element. solidProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for \_Solid.

#### 10.2.3.3 gml:SolidArrayPropertyType, gml:solidArrayProperty

```
<complexType name="SolidArrayPropertyType">  
  <sequence>  
    <element ref="gml:_Solid" minOccurs="0" maxOccurs="unbounded" />  
  </sequence>  
</complexType>
```

A container for an array of solids. The elements are always contained in the array property, referencing geometry elements or arrays of geometry elements is not supported.

```
<element name="solidArrayProperty" type="gml:SolidArrayPropertyType" />
```

This property element contains a list of solid elements. solidArrayProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for a list of \_Solid.

#### 10.2.3.4 gml:SolidType, gml:Solid

```
<complexType name="SolidType">  
  <complexContent>  
    <extension base="gml:AbstractSolidType">  
      <sequence>  
        <element name="exterior" type="gml:SurfacePropertyType" minOccurs="0" />  
        <element name="interior" type="gml:SurfacePropertyType" minOccurs="0"  
               maxOccurs="unbounded" />  
      </sequence>  
    </extension>  
  </complexContent>  
</complexType>  
  
<element name="Solid" type="gml:SolidType" substitutionGroup="gml:_Solid" />
```

A solid is the basis for 3-dimensional geometry. The extent of a solid is defined by the boundary surfaces (shells). A shell is represented by a composite surface, where every shell is used to represent a single connected component of the boundary of a solid. It consists of a composite surface (a list of orientable surfaces) connected in a topological cycle (an object whose boundary is empty). Unlike a Ring, a shell's elements have no natural sort order. Like Rings, shells are simple.

The element "exterior" specifies the outer boundary of the solid. Boundaries of solids are similar to surface boundaries. In normal 3-dimensional Euclidean space, one (composite) surface is distinguished as the exterior. In the more general case, this is not always possible.

The element "interior" specifies the inner boundary of the solid. Boundaries of solids are similar to surface boundaries.

## 11 GML schemas – geometric complex, geometric composites and geometric aggregates

### 11.1 Overview

This clause describes the geometry model used by GML in the XML Schema documents:

- geometryAggregates.xsd
- geometryComplexes.xsd

The GML types and elements are described in the schemas listed in Annex C. The geometry schemas are identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:geometryAggregates:v3.1.0  
 urn:opengis:specification:gml:schema-xsd:geometryComplexes:v3.1.0

All types, elements and attributes in these XML Schema documents are covered by this Clause.

Geometric aggregates (i.e. instances of a subtype of AbstractGeometricAggregateType) are arbitrary aggregations of geometry elements. They are not assumed to have any additional internal structure and are used to "collect" pieces of geometry of a specified type. Application schemas may use aggregates for features that use multiple geometric objects in their representations.

Geometric complexes (i.e. instances of GeometricComplexType) are closed collections of geometric primitives, i.e. they will contain their boundaries. The composite geometries (CompositeCurve, CompositeSurface and CompositeSolid) can be viewed as primitives and as complexes. See Clause 7.6 and ISO 19107 for more details on the nature of composite geometries.

### 11.2 Geometric complex and geometric composites

A geometric complex is a set of primitive geometric objects (in a common coordinate system) whose interiors are disjoint. Further, if a primitive is in a geometric complex, then there exists a set of primitives in that complex whose point-wise union is the boundary of this first primitive.

A GeometricComplex in GML is a collection of geometrically disjoint, simple primitives. If a geometric primitive (other than a Point) is in a particular GeometricComplex, then there exists a set of primitives of lower dimension in the same complex that form the boundary of this primitive.

A geometric composite (GML specifies CompositeCurve, CompositeSurface or CompositeSolid) represents a geometric complex with an underlying core geometry that is isomorphic to a primitive. Thus, a composite curve is a collection of curves whose geometry could be viewed as a curve (albeit one with a complex inner

structure). Composites are intended for use as attribute values in datasets in which the underlying geometry has been decomposed, usually to expose its topological nature.

The members of a geometric composite shall represent a homogeneous collection of geometric primitives whose union would be the core geometry of the composite. The complex would include all member primitives and all primitives on the boundary of these primitives, and so forth until Points are included. Thus the „member“ properties in CompositeCurve, CompositeSurface and CompositeSolid represent a subset of the „element“ property of GeometricComplex.

Geometric complexes and composites shall be used in application schemas where the sharing of geometry is important.

For more information about geometric complexes and an extensive discussion about complexes/composites/primitives and their commonalities/differences see ISO 19107.

As XML Schema does not support the concept of “multiple inheritance” which is used in ISO 19107 to express the duality of the geometric composites (as an open primitive and as a closed complex) in the GML geometry model, the composites inherit from AbstractGeometricPrimitiveType only. However, by using a `<choice>` element, a composite can be used in any property, which expects a GeometricComplex as its value.

### 11.2.1 Composite Geometries

#### 11.2.1.1 `gml:CompositeCurveType`, `gml:CompositeCurve`

```
<complexType name="CompositeCurveType">
  <complexContent>
    <extension base="gml:AbstractCurveType">
      <sequence>
        <element ref="gml:curveMember" maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="CompositeCurve" type="gml:CompositeCurveType" substitutionGroup="gml:_Curve" />
```

A CompositeCurve is defined by a sequence of (orientable) curves such that the each curve in the sequence terminates at the start point of the subsequent curve in the list.

The element “curveMember” references or contains one curve in the composite curve.

The curves are contiguous, the collection of curves is ordered.

NOTE This definition allows for a nested structure, i.e. a CompositeCurve may use, for example, another CompositeCurve as a curve member.

#### 11.2.1.2 `gml:CompositeSurfaceType`, `gml:CompositeSurface`

```
<complexType name="CompositeSurfaceType">
  <complexContent>
    <extension base="gml:AbstractSurfaceType">
      <sequence>
        <element ref="gml:surfaceMember" maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="CompositeSurface" type="gml:CompositeSurfaceType" substitutionGroup="gml:_Surface" />
```

A CompositeSurface is defined by a set of orientable surfaces. A composite surface is geometry type with all the geometric properties of a (primitive) surface. Essentially, a composite surface is a collection of surfaces that join in pairs on common boundary curves and which, when considered as a whole, form a single surface.

The element “surfaceMember” references or contains one surface in the composite surface. The surfaces are contiguous.

NOTE This definition allows for a nested structure, i.e. a CompositeSurface may use, for example, another CompositeSurface as a member.

### 11.2.1.3 **gml:CompositeSolidType, gml:CompositeSolid**

```
<complexType name="CompositeSolidType">
  <complexContent>
    <extension base="gml:AbstractSolidType">
      <sequence>
        <element ref="gml:solidMember" maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="CompositeSolid" type="gml:CompositeSolidType" substitutionGroup="gml:_Solid" />
```

A composite solid is a geometry type with all the geometric properties of a (primitive) solid.

Essentially, a composite solid is a collection of solids that join in pairs on common boundary surfaces and which, when considered as a whole, form a single solid.

The element “generator” references or contains one solid in the composite solid. The solids are contiguous.

NOTE This definition allows for a nested structure, i.e. a CompositeSolid may use, for example, another CompositeSolid as a member.

## 11.2.2 Geometric Complex

### 11.2.2.1 **gml:GeometricComplexType, gml:GeometricComplex**

```
<complexType name="GeometricComplexType">
  <complexContent>
    <extension base="gml:AbstractGeometryType">
      <sequence>
        <element name="element" type="gml:GeometricPrimitivePropertyType"
          maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="GeometricComplex" type="gml:GeometricComplexType" substitutionGroup="gml:_Geometry" />
```

A geometric complex. Every element “element” references or contains one geometric primitive (this includes composite geometries).

### 11.2.2.2 **gml:GeometricComplexPropertyType**

```
<complexType name="GeometricComplexPropertyType">
  <choice minOccurs="0">
    <element ref="gml:GeometricComplex" />
    <element ref="gml:CompositeCurve" />
    <element ref="gml:CompositeSurface" />
    <element ref="gml:CompositeSolid" />
```

```

</choice>
<attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>

```

A property that has a geometric complex as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

**NOTE** The allowed geometry elements contained in such a property (or referenced by it) have to be modeled by an XML Schema choice element since the composites (conceptually) inherit both from geometric complex *and* geometric primitive and are already part of the \_GeometricPrimitive substitution group.

## 11.3 Geometric Aggregates

### 11.3.1 Aggregates of unspecified dimensionality

#### 11.3.1.1 gml:AbstractGeometricAggregateType, gml:\_GeometricAggregate

```

<complexType name="AbstractGeometricAggregateType" abstract="true">
    <complexContent>
        <extension base="gml:AbstractGeometryType" />
    </complexContent>
</complexType>

```

This is the abstract root type of the geometric aggregates.

```

<element name="_GeometricAggregate" type="gml:AbstractGeometricAggregateType" abstract="true"
substitutionGroup="gml:_Geometry" />

```

The "\_GeometricAggregate" element is the abstract head of the substitution group for all geometric aggregates.

#### 11.3.1.2 gml:MultiGeometryType, gml:MultiGeometry, gml:geometryMember, gml:geometryMembers

```

<complexType name="MultiGeometryType">
    <complexContent>
        <extension base="gml:AbstractGeometricAggregateType">
            <sequence>
                <element ref="gml:geometryMember" minOccurs="0" maxOccurs="unbounded" />
                <element ref="gml:geometryMembers" minOccurs="0" />
            </sequence>
        </extension>
    </complexContent>
</complexType>

<element name="MultiGeometry" type="gml:MultiGeometryType" substitutionGroup="gml:_GeometricAggregate" />

```

A geometry collection shall include one or more geometries, referenced through geometryMember elements.

The members of the geometric aggregate can be specified either using the "standard" property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection.

**NOTE** Array properties cannot reference remote geometry elements via XLinks.

```

<element name="geometryMember" type="gml:GeometryPropertyType" />

```

This property element either references a geometry element via the XLink-attributes or contains the geometry element.

```

<element name="geometryMembers" type="gml:GeometryArrayPropertyType" />

```

This property element contains a list of geometry elements. The order of the elements is significant and shall be preserved when processing the array.

### 11.3.1.3 **gml:MultiGeometryPropertyType, gml:multiGeometryProperty**

```
<complexType name="MultiGeometryPropertyType">
  <sequence>
    <element ref="gml:_GeometricAggregate" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

A property that has a geometric aggregate as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

```
<element name="multiGeometryProperty" type="gml:MultiGeometryPropertyType" />
```

This property element either references a geometric aggregate via the XLink-attributes or contains the "multi geometry" element. multiGeometryProperty is the predefined property, which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for \_GeometricAggregate.

### 11.3.2 0-dimensional aggregates

#### 11.3.2.1 **gml:MultiPointType, gml:MultiPoint, gml:pointMember, gml:pointMembers**

```
<complexType name="MultiPointType">
  <complexContent>
    <extension base="gml:AbstractGeometricAggregateType">
      <sequence>
        <element ref="gml:pointMember" minOccurs="0" maxOccurs="unbounded" />
        <element ref="gml:pointMembers" minOccurs="0" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="MultiPoint" type="gml:MultiPointType" substitutionGroup="gml:_GeometricAggregate" />
```

A MultiPoint is defined by one or more Points, referenced through pointMember elements.

The members of the geometric aggregate can be specified either using the "standard" property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection.

**NOTE** Array properties cannot reference remote geometry elements via XLinks.

```
<element name="pointMember" type="gml:PointPropertyType" />
```

This property element either references a Point via the XLink-attributes or contains the Point element.

```
<element name="pointMembers" type="gml:PointArrayPropertyType" />
```

This property element contains a list of points. The order of the elements is significant and shall be preserved when processing the array.

#### 11.3.2.2 **gml:MultiPointPropertyType, gml:multiPointProperty**

```
<complexType name="MultiPointPropertyType">
  <sequence>
```

```

        <element ref="gml:MultiPoint" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

A property that has a collection of points as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

```
<element name="multiPointProperty" type="gml:MultiPointPropertyType" />
```

This property element either references a point aggregate via the XLink-attributes or contains the "multi point" element. multiPointProperty is the predefined property, which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for MultiPoint.

### 11.3.3 1-dimensional aggregates

#### 11.3.3.1 gml:MultiCurveType, gml:multiCurve, gml:curveMembers

```

<complexType name="MultiCurveType">
    <complexContent>
        <extension base="gml:AbstractGeometricAggregateType">
            <sequence>
                <element ref="gml:curveMember" minOccurs="0" maxOccurs="unbounded" />
                <element ref="gml:curveMembers" minOccurs="0" />
            </sequence>
        </extension>
    </complexContent>
</complexType>

<element name="MultiCurve" type="gml:MultiCurveType" substitutionGroup="gml:_GeometricAggregate" />
```

A MultiCurve is defined by one or more Curves, referenced through curveMember elements.

The members of the geometric aggregate can be specified either using the "standard" property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection.

NOTE      Array properties cannot reference remote geometry elements via XLinks.

```
<element name="curveMembers" type="gml:CurveArrayPropertyType" />
```

This property element contains a list of curves. The order of the elements is significant and shall be preserved when processing the array.

#### 11.3.3.2 gml:MultiCurvePropertyType, gml:multiCurveProperty

```

<complexType name="MultiCurvePropertyType">
    <sequence>
        <element ref="gml:MultiCurve" minOccurs="0" />
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

A property that has a collection of curves as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

```
<element name="multiCurveProperty" type="gml:MultiCurvePropertyType" />
```

This property element either references a curve aggregate via the XLink-attributes or contains the "multi curve" element. multiCurveProperty is the predefined property, which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for MultiCurve.

### 11.3.3.3 **gml:lineStringMember**

```
<element name="lineStringMember" type="gml:LineStringPropertyType" />
```

*Deprecated with GML 3.0 and included only for backwards compatibility with GML 2. Use "curveMember" instead.*

*This property element either references a line string via the XLink-attributes or contains the line string element.*

### 11.3.3.4 **gml:MultiLineStringType, gml:MultiLineString**

```
<complexType name="MultiLineStringType">
  <complexContent>
    <extension base="gml:AbstractGeometricAggregateType">
      <sequence>
        <element ref="gml:lineStringMember" minOccurs="0" maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A MultiLineString is defined by one or more LineStrings, referenced through lineStringMember elements. *Deprecated with GML version 3.0. Use MultiCurveType instead.*

```
<element name="MultiLineString" type="gml:MultiLineStringType" substitutionGroup="gml:_GeometricAggregate" />
```

*Deprecated with GML 3.0 and included for backwards compatibility with GML 2. Use the "MultiCurve" element instead.*

### 11.3.3.5 **gml:MultiLineStringPropertyType**

```
<complexType name="MultiLineStringPropertyType">
  <sequence>
    <element ref="gml:MultiLineString" minOccurs="0" />
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

*This type is deprecated with GML 3.0 and shall not be used. It is included for backwards compatibility with GML 2. Use MultiCurvePropertyType instead.*

A property that has a collection of line strings as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

## 11.3.4 2-dimensional aggregates

### 11.3.4.1 **gml:MultiSurfaceType, gml:MultiSurface, gml:surfaceMember, gml:surfaceMembers**

```
<complexType name="MultiSurfaceType">
  <complexContent>
    <extension base="gml:AbstractGeometricAggregateType">
      <sequence>
        <element ref="gml:surfaceMember" minOccurs="0" maxOccurs="unbounded" />
        <element ref="gml:surfaceMembers" minOccurs="0" />
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

</sequence>
</extension>
</complexContent>
</complexType>

<element name="MultiSurface" type="gml:MultiSurfaceType" substitutionGroup="gml:_GeometricAggregate" />

```

A MultiSurface is defined by one or more Surfaces, referenced through surfaceMember elements.

The members of the geometric aggregate can be specified either using the "standard" property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection.

NOTE Array properties cannot reference remote geometry elements via XLinks.

```
<element name="surfaceMember" type="gml:SurfacePropertyType" />
```

This property element either references a surface via the XLink-attributes or contains the surface element. A surface element is any element, which is substitutable for "\_Surface".

```
<element name="surfaceMembers" type="gml:SurfaceArrayPropertyType" />
```

This property element contains a list of surfaces. The order of the elements is significant and shall be preserved when processing the array.

#### 11.3.4.2 gml:MultiSurfacePropertyType, gml:multiSurfaceProperty

```

<complexType name="MultiSurfacePropertyType">
  <sequence>
    <element ref="gml:MultiSurface" minOccurs="0" />
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>

```

A property that has a collection of surfaces as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

```
<element name="multiSurfaceProperty" type="gml:MultiSurfacePropertyType" />
```

This property element either references a surface aggregate via the XLink-attributes or contains the "multi surface" element. multiSurfaceProperty is the predefined property, which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for MultiSurface.

#### 11.3.4.3 gml:MultiPolygonType, gml:MultiPolygon, gml:polygonMember

```

<complexType name="MultiPolygonType">
  <complexContent>
    <extension base="gml:AbstractGeometricAggregateType">
      <sequence>
        <element ref="gml:polygonMember" minOccurs="0" maxOccurs="unbounded" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

A MultiPolygon is defined by one or more Polygons, referenced through polygonMember elements. Deprecated with GML version 3.0. Use MultiSurfaceType instead.

```
<element name="MultiPolygon" type="gml:MultiPolygonType" substitutionGroup="gml:_GeometricAggregate" />
```

*Deprecated with GML 3.0 and included for backwards compatibility with GML 2. Use the "MultiSurface" element instead.*

```
<element name="polygonMember" type="gml:PolygonPropertyType" />
```

*Deprecated with GML 3.0 and included only for backwards compatibility with GML 2.0. Use "surfaceMember" instead.*

*This property element either references a polygon via the XLink-attributes or contains the polygon element.*

#### 11.3.4.4 gml:MultiPolygonPropertyType

```
<complexType name="MultiPolygonPropertyType">
  <sequence>
    <element ref="gml:MultiPolygon" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

*This type is deprecated with GML 3.0 and shall not be used. It is included for backwards compatibility with GML 2. Use MultiSurfacePropertyType instead.*

*A property that has a collection of polygons as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.*

#### 11.3.5 3-dimensional aggregates

##### 11.3.5.1 gml:MultiSolidType, gml:MultiSolid, gml:solidMember, gml:solidMembers

```
<complexType name="MultiSolidType">
  <complexContent>
    <extension base="gml:AbstractGeometricAggregateType">
      <sequence>
        <element ref="gml:solidMember" minOccurs="0" maxOccurs="unbounded" />
        <element ref="gml:solidMembers" minOccurs="0" />
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="MultiSolid" type="gml:MultiSolidType" substitutionGroup="gml:_GeometricAggregate" />
```

A MultiSolid is defined by one or more Solids, referenced through solidMember elements.

The members of the geometric aggregate can be specified either using the "standard" property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection.

NOTE      Array properties cannot reference remote geometry elements via XLinks.

```
<element name="solidMember" type="gml:SolidPropertyType" />
```

This property element either references a solid via the XLink-attributes or contains the solid element. A solid element is any element, which is substitutable for "\_Solid".

```
<element name="solidMembers" type="gml:SolidArrayPropertyType" />
```

This property element contains a list of solids. The order of the elements is significant and shall be preserved when processing the array.

### 11.3.5.2 **gml:MultiSolidPropertyType**, **gml:multiSolidProperty**

```
<complexType name="MultiSolidPropertyType">
  <sequence>
    <element ref="gml:MultiSolid" minOccurs="0" />
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>
```

A property that has a collection of solids as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element shall be given, but neither both nor none.

```
<element name="multiSolidProperty" type="gml:MultiSolidPropertyType" />
```

This property element either references a solid aggregate via the XLink-attributes or contains the "multi solid" element. multiSolidProperty is the predefined property, which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for MultiSolid.

## 12 GML schemas – Coordinate reference systems schemas

### 12.1 Overview

#### 12.1.1 Introduction

This clause describes the six GML schema documents for encoding the definitions of coordinate reference systems and coordinate operations, explaining their contents, structure, and dependencies. Those six schema documents are:

- a) referenceSystems.xsd
- b) coordinateReferenceSystems.xsd
- c) datums.xsd
- d) coordinateSystems.xsd
- e) coordinateOperations.xsd
- f) dataQuality.xsd

All these schema documents are closely based on the UML package with a similar name in OGC Abstract Specification Topic 2, which is based on ISO 19111. All these schema documents are listed in Annex C. All these schema documents describe components in the <http://www.opengis.net/gml> namespace, for which the prefix **gml** is normally used.

#### 12.1.2 Coordinates

The position of a point can be described by (a set of) coordinates. Coordinates are unambiguous only when the associated coordinate reference system has been fully defined.

The geometry of spatial features might alternately be expressed in terms of invariant geometric quantities, viz. shapes and relative positions/orientations (strictly speaking only distance ratios and angles are invariant quantities). However, this would be unworkable: performing calculations on spatial data would become a major effort. The expression of the position of a point by coordinates introduces simplicity in terms of overview and calculus.

However, there is a price to be paid for this convenience. To describe a simple shape such as a triangle in a plane, six (plane) coordinates are required, whereas only a single distance ratio and an angle would suffice. The inherent degrees of freedom (four in 2D, seven in 3D) have to be satisfied by choosing the origin of the coordinate axes, their unit of measure, and the orientations of the axes. This choice underlines the fact that coordinates are human-defined quantities and not natural phenomena. Although this may seem self-evident, it is often overlooked and has consequences for the interpretation of coordinates and their error characteristics.

The choice of values for the parameters that constitute the degrees of freedom of the coordinate space is captured in the concept of a *coordinate reference system*. Without the full specification of the coordinate reference system, coordinates are ambiguous at best and meaningless at worst. The fact that such a choice *must* be made, either arbitrarily or by adopting values from survey measurements, leads to the large number of coordinate reference systems in use around the world. It also causes the little understood fact that the latitude and longitude of a point are not unique.

In this document, the term “coordinates” means the tuple of ordered scalar values that define the position of a single point in a coordinate reference system. The order of the values in the tuple and their unit(s) of measure are parts of the coordinate reference system definition. The tuple is composed of one, two, or three “ordinates”. The ordinates must be mutually independent and their number must be equal to the dimension of the coordinate space; for example, a tuple of coordinates may not contain two heights. In this document, the term “set of coordinates” is used to indicate the coordinates of multiple points.

However, it must be pointed out that usage in practice of the term “coordinates” is subject to ambiguity. Sometimes the term “coordinate” is used to indicate the tuple; the plural “coordinates” in that case describes the coordinates of multiple points. Others use the term “set of coordinates” to describe the coordinates of a single point. In that case the term “coordinate” is sometimes used instead of “ordinate”. The reader is advised to carefully infer the intended meaning from the context.

The concept of coordinates may be expanded from a strictly spatial context to include time. Time is then added as another ordinate to the coordinate tuple. It is even possible to add two time-coordinates, provided the two coordinates describe different independent quantities. An example of the latter is the time/space position of a subsurface point feature of which the vertical ordinate is expressed as the two-way travel time of a sound signal in milliseconds, as is common in seismic imaging and a second time ordinate indicates the time of observation, usually expressed in whole years.

In summary: each tuple of coordinates describing the position of a point shall be associated with a coordinate reference system. Instead of supplying the definition of the coordinate reference system with every single point, coordinates are often supplied in data sets in which all coordinates belong to the same coordinate reference system. Each data set then uses one coordinate reference system definition or reference, which applies to all coordinates in that data set.

### 12.1.3 Some geodetic concepts

Geodesy is the applied science that aims to determine the size and shape of the earth. In a more practical and local sense, this may be understood to mean the determination of the relative positions of points on or near the earth's surface. Survey measurements and techniques are the means used to achieve this aim.

The most accurate reference shape approximating the earth is the geoid, the surface that is defined as the locus of all points with equal gravity at mean sea level. This shape excludes topography and the effects of tides, currents and weather on the oceans and seas. Topographic heights are typically expressed relative to the geoid. The gravity vector at mean sea level is everywhere perpendicular to this surface. Due to the irregular mass distribution in the earth's interior, the geoid has an irregular shape. This makes the geoid unsuitable to use in calculations on spatial data; unfortunate because the familiar concept of height derives from the geoid and the geoid therefore implicitly plays an important role in all engineering and mapping activities. The direction of gravity also plays an important role in the mentioned survey techniques.

The geoid is approximated by the nearest regular body, an oblate spheroid, of which the oblateness corresponds to the flattening of the physical earth (and thus the geoid) at the poles due to the earth's rotation. In survey practice, this spheroid is often referred to as an ellipsoid. Although mathematically the term “spheroid” is a more precise description than “ellipsoid”, the latter term is used in preference of “spheroid”, in

accordance with ISO 19111. The ellipsoid is a reasonably accurate approximation of the geoid, the latter undulating around the ellipsoid's surface with variations only in the order of several tens of metres.

The advantage of the ellipsoid is that it is much easier to work with mathematically than the geoid. The ellipsoid forms the basis of the best-known type of coordinate reference systems: the Geographic CRS. The position of a point relative to the ellipsoid is expressed by means of geographic coordinates: geodetic latitude ( $\phi$ ) and geodetic longitude ( $\lambda$ ). The height ( $h$ ) above the ellipsoid is of not much practical use because everyday heights are related to the geoid (gravity-related heights, indicated by  $H$ ). Ellipsoidal height is an inseparable element of a 3D coordinate tuple, and originates either directly from a 3D survey technique (e.g. GPS) or from the transformation of horizontal coordinates extended with a gravity-related height.

Unfortunately there is not just one ellipsoid. An ellipsoid is a matter of choice, and therefore many choices are possible. The size and shape of the ellipsoid are traditionally chosen such that the surface of the geoid is matched as closely as possible locally, e.g. in a country, although a number global of best-fits are available. Each association of an ellipsoid with earth surface geometry results in the implicit choice of parameters to satisfy the degrees of freedom problem described in Subclause 12.1.2 above. In this case, the choice results in the definition of the origin, orientation, size, and shape of the ellipsoid. Collectively this choice is captured by the concept of "geodetic datum".

A Geographic CRS is not suitable for mapmaking on a planar surface, because it describes positions on a curved surface. It is impossible to represent such geometry in a Euclidean plane without introducing distortions. The need to control these distortions has given rise to the development of the science of map projections. Although some map projections can be represented as a geometric process, in general a map projection is a set of formulae that converts geodetic latitude and longitude to plane (map) coordinates. Height plays no role in this process, which is entirely two-dimensional.

Heights are traditionally determined separate from horizontal position in separate height networks, which is due to the different survey techniques used for the determination of horizontal and vertical geometry. Most practical heights are defined in close relationship with the gravity vector. Geodetic science distinguishes several different types of gravity-related heights. The differences between those are considered irrelevant for the purposes of this specification.

#### **12.1.4 Non-geodetic coordinate reference systems**

##### **12.1.4.1 Introduction**

In addition to the geographic and related coordinate reference systems discussed above, several other types of CRSs are when needed, and are handled by the GML schemas discussed. These other types of CRSs include:

##### **12.1.4.2 Geocentric CRS**

Type of coordinate reference system that deals with the earth's curvature by taking the 3D spatial view, which obviates the need to model the earth's curvature. The origin of a geocentric CRS is at the approximate centre of mass of the earth.

##### **12.1.4.3 Temporal CRS**

Used for the recording of time in association with any of the listed spatial coordinate reference systems.

##### **12.1.4.4 Engineering CRS**

Type of coordinate reference system that is used in a contextually local sense. This sub-type is used to model two broad categories of local coordinate reference systems:

- a) earth-fixed systems, applied to activities on or near the surface of the earth;

- b) spatial referencing by coordinates on moving platforms such as road vehicles, vessels, aircraft or spacecraft.

Earth-fixed Engineering CRSs are commonly based on a simple flat-earth approximation of the earth's surface, and the effect of earth curvature on feature geometry is ignored: calculations on coordinates use simple plane arithmetic without any corrections for earth curvature. The application of such Engineering CRSs to relatively small areas and "contextually local" is in this case equivalent to "spatially local".

One specialized type of earth-fixed Engineering CRS is a linear CRS, which represents position along a specified (usually not straight) line.

Engineering CRSs used on moving platforms are usually intermediate coordinate reference systems that are computationally required to calculate geodetic coordinates. These coordinate reference systems are subject to all the motions of the platform with which they are associated. In this case "contextually local" means that the associated coordinates are meaningful only relative to the moving platform. Earth curvature is usually irrelevant and is therefore ignored. In the spatial sense their applicability may extend from the immediate vicinity of the platform (e.g. a moving seismic ship) to the entire earth (e.g. in space applications). The determining factor is the mathematical model deployed in the positioning calculations. Transformation of coordinates from these moving Engineering CRSs to earth-referenced coordinate reference systems involves time-dependent coordinate operation parameters, which can be modelled by the structures provided in this UML model.

#### 12.1.4.5 Image CRS

An Image CRS is an Engineering CRS applied to images. Image CRSs warrant treatment as a separate subtype because a separate user community exists for images with its own vocabulary. The definition of the associated Image Datum contains two data quantities not relevant for other datums and coordinate reference systems.

#### 12.1.4.6 Derived CRS

Some coordinate reference systems are defined by applying a coordinate conversion to another coordinate reference system. Such a coordinate reference system is called a Derived CRS, and the coordinate reference system it was derived from by applying the conversion is called the Source CRS. A coordinate conversion is an arithmetic operation with zero or more parameters that have defined values. The best-known example of a Derived CRS is a Projected CRS, which is always derived from a source Geographic CRS by applying the coordinate conversion known as a map projection.

An example of a Derived CRS of *derivedCRSType*: "geographic" is one of which the unit of measure has been modified with respect to an earlier defined Geographic CRS, which then takes the role of Source CRS. Another example is the CRS for a set of gridded data, where row and column coordinates reflect the position in the grid.

#### 12.1.5 Important XML elements

These XML Schemas encode definition data for both Coordinate Reference Systems (CRSs) and Coordinate Operations (including coordinate Transformations and Conversions). This definition data includes identification and specification data, both included as needed. The definition data for a CRS includes or references definition data for a Coordinate System and a Datum, as appropriate. When applicable, definition data for a CRS includes or references definition data for a Coordinate Operation, and vice-versa.

The specified XML encoding includes multiple alternative top level XML elements that can be used where needed. (That is, there is not a single top level element that may be the basis for all XML documents.) Most of these top level XML elements are GML objects that include identification information, allowing it to be referenced. The alternative top level XML elements include:

- a) All concrete XML elements in the substitution group headed by the abstract *CoordinateReferenceSystem* XML element. These elements can each be used to transfer the definition of one coordinate reference system of that type. These nine concrete XML elements are named:

- 1) CompoundCRS
  - 2) GeocentricCRS
  - 3) GeographicCRS
  - 4) ProjectedCRS
  - 5) EngineeringCRS
  - 6) ImageCRS
  - 7) VerticalCRS
  - 8) TemporalCRS
  - 9) DerivedCRS
- b) All concrete XML elements that are substitutable for the abstract CoordinateOperation XML element, namely:
- 1) ConcatenatedOperation. This element should be used to encode the definition of a concatenated coordinate conversion, which combines a specified sequence of Transformations and/or Conversions.
  - 2) PassThroughOperation. This element should be used to encode the definition of a pass through coordinate conversion, which uses a specified Conversion or Transformation to convert a subset of the coordinates of a CompoundCRS.
  - 3) Transformation. This element may be used to encode the definition of an approximate coordinate transformation, from one specified source coordinate reference system to another specified target coordinate reference system.
  - 4) Conversion. This element may be used to encode the definition of an exact coordinate conversion, from one coordinate reference system to another coordinate reference system.

NOTE 1 The Conversion and Transformation elements should not be used for well-known coordinate operation methods with many instances.

The concrete XML elements that are substitutable for CoordinateReferenceSystem use multiple lower level XML elements containing data structures. These lower level elements include all five concrete elements that are substitutable for the abstract Datum XML element, named:

- a) GeodeticDatum
- b) VerticalDatum
- c) TemporalDatum
- d) EngineeringDatum
- e) ImageDatum

These lower level XML elements also include all ten concrete elements that are substitutable for the abstract CoordinateSystem XML element, named:

- a) EllipsoidalCS

- b) VerticalCS
- c) CartesianCS
- d) ObliqueCartesianCS
- e) LinearCS
- f) PolarCS
- g) SphericalCS
- h) CylindricalCS
- i) TemporalCS
- j) UserDefinedCS

The concrete XML elements that are substitutable for the CoordinateOperation element use multiple lower level elements containing data structures, including the elements named:

- a) CovarianceMatrix
- b) CovarianceElement
- c) OperationMethod
- d) OperationParameter
- e) OperationParameterGroup
- f) ParameterValue
- g) ParameterValueGroup

NOTE 2 The ParameterValue and ParameterValueGroup elements should not be used for well-known coordinate operation methods with many instances.

## 12.2 referenceSystems schema

### 12.2.1 Overview

The reference systems schema, referenceSystems.xsd, has three logical parts, which define elements and types for XML encoding of the definitions of:

- a) Object identification, of the ten types of GML objects used for coordinate reference systems and coordinate operations
- b) Validity, of three types of GML objects
- c) High-level part of the definitions of coordinate reference systems

This schema encodes the Reference System package of the UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex B of Topic 2. The SC\_CRS class is also encoded here, to eliminate the (circular) references from coordinateOperations.xsd to coordinateReferenceSystems.xsd. The RS\_SpatialReferenceSystemUsingGeographicIdentifier class is not encoded, since it is not applicable to coordinate positions. The CI\_Citation class is not directly encoded, since such information can be included as

metaDataProperty elements which are optionally allowed. A modified version of the EX\_Extent (DataType) class from ISO 19115 is currently encoded here, using GML 3 schema types.

The referenceSystems schema includes the dictionary.xsd, geometryBasic2d.xsd, and temporal.xsd GML schemas. This schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:referenceSystems:v3.1.0

## 12.2.2 Identification

### 12.2.2.1 gml:IdentifierType

```
<complexType name="IdentifierType">
  <sequence>
    <element ref="gml:name"/>
    <element ref="gml:version" minOccurs="0"/>
    <element ref="gml:remarks" minOccurs="0">
      </element>
    </sequence>
  </complexType>
```

An identification of a CRS object. The first use of the IdentifierType for an object, if any, is normally the primary identification code, and any others are aliases.

The IdentifierType uses gml:name for the Identifier code or name, often from a controlled list or pattern defined by a code space. The optional codeSpace attribute is normally included to identify or reference a code space within which one or more codes are defined. This code space is often defined by some authority organization, where one organization may define multiple code spaces. The range and format of each Code Space identifier is defined by that code space authority. Information about that code space authority can be included as metaDataProperty elements which are optionally allowed in all CRS objects.

### 12.2.2.2 gml:version

```
<element name="version" type="string"/>
```

Identifier of the version of the associated codeSpace or code, as specified by the codeSpace or code authority. This version is included only when the "code" or "codeSpace" uses versions. When appropriate, the version is identified by the effective date, coded using ISO 8601 date format.

### 12.2.2.3 gml:remarks

```
<element name="remarks" type="gml:StringOrRefType" substitutionGroup="gml:description"/>
```

Information about this code or alias. Contains text or refers to external text.

### 12.2.2.4 gml:SimpleNameType

```
<complexType name="SimpleNameType">
  <simpleContent>
    <restriction base="gml:CodeType">
      <attribute name="codeSpace" type="anyURI" use="prohibited"/>
    </restriction>
  </simpleContent>
</complexType>
```

The primary name of an object. The string in the CodeType contains the object identification name, and the codeSpace attribute is not included.

### 12.2.2.5 gml:scope

```
<element name="scope" type="string"/>
```

Description of the domain of usage, or limitations of usage, for which this CRS object is valid.

## 12.2.3 Validity

### 12.2.3.1 gml:validArea

```
<element name="validArea" type="gml:ExtentType"/>
```

Area or region in which this CRS object is valid

### 12.2.3.2 gml:ExtentType

```
<complexType name="ExtentType">
  <sequence>
    <element ref="gml:description" minOccurs="0"/>
    <choice>
      <element ref="gml:boundingBox" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:boundingPolygon" minOccurs="0" maxOccurs="unbounded"/>
    </choice>
    <element ref="gml:verticalExtent" minOccurs="0" maxOccurs="unbounded"/>
    <element ref="gml:temporalExtent" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
```

Information about the spatial, vertical, and/or temporal extent of a reference system object. Constraints: At least one of the elements "description", "boundingBox", "boundingPolygon", "verticalExtent", and "temporalExtent" shall be included, but more than one can be included when appropriate. Furthermore, more than one "boundingBox", "boundingPolygon", "verticalExtent", and/or "temporalExtent" element can be included, with more than one meaning the union of the individual domains.

The "description" element contains a description of spatial and/or temporal extent of this object. The boundingBox and boundingPolygon are alternative elements for defining the Geographic domain of this reference system object. If the "srsName" value for the boundingBox element references a CRS that includes elevation and time, the verticalExtent and temporalExtent elements may not be needed.

The boundingBox elements are an unordered list of bounding boxes (or envelopes) whose union describes the spatial domain of this object. The boundingPolygon elements are an unordered list of bounding polygons whose union describes the spatial domain of this object. The verticalExtent elements are an unordered list of vertical intervals whose union describes the spatial domain of this object. The temporalExtent elements are an unordered list of time periods whose union describes the spatial domain of this object.

### 12.2.3.3 gml:boundingBox

```
<element name="boundingBox" type="gml:EnvelopeType"/>
```

A bounding box (or envelope) defining the spatial domain of this object.

### 12.2.3.4 gml:boundingPolygon

```
<element name="boundingPolygon" type="gml:PolygonType"/>
```

A bounding polygon horizontal defining the horizontal spatial domain of this object.

### 12.2.3.5 gml:verticalExtent

```
<element name="verticalExtent" type="gml:EnvelopeType"/>
```

An interval defining the vertical spatial domain of this object.

#### 12.2.3.6 **gml:temporalExtent**

```
<element name="temporalExtent" type="gml:TimePeriodType"/>
```

A time period defining the temporal domain of this object.

### 12.2.4 High-level coordinate reference system

#### 12.2.4.1 **gml:\_ReferenceSystem**

```
<element name="_ReferenceSystem" type="gml:AbstractReferenceSystemType" abstract="true"
substitutionGroup="gml:Definition"/>

<complexType name="AbstractReferenceSystemBaseType" abstract="true">
<complexContent>
<restriction base="gml:DefinitionType">
<sequence>
<element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:remarks" minOccurs="0">
<element ref="gml:srsName"/>
</sequence>
<attribute ref="gml:id" use="required"/>
</restriction>
</complexContent>
</complexType>

<complexType name="AbstractReferenceSystemType" abstract="true">
<complexContent>
<extension base="gml:AbstractReferenceSystemBaseType">
<sequence>
<element ref="gml:srsID" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:validArea" minOccurs="0"/>
<element ref="gml:scope" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>
```

Description of a spatial and/or temporal reference system used by a dataset.

The AbstractReferenceSystemBaseType provides basic encoding for reference system objects, simplifying and restricting the DefinitionType as needed.

The "remarks" element contains comments on or information about this reference system, including source information.

The "srsID" elements contain a set of alternative identifications of this reference system. The first srsID, if any, is normally the primary identification code, and any others are aliases.

#### 12.2.4.2 **gml:srsName**

```
<element name="srsName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which this reference system is identified.

#### 12.2.4.3 **gml:srsID**

```
<element name="srsID" type="gml:IdentifierType"/>
```

An identification of a reference system.

#### 12.2.4.4 **gml:referenceSystemRef**

```
<element name="referenceSystemRef" type="gml:ReferenceSystemRefType"
substitutionGroup="gml:dictionaryEntry"/>

<complexType name="ReferenceSystemRefType">
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:_ReferenceSystem" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a reference system, either referencing or containing the definition of that reference system.

#### 12.2.4.5 **gml:\_CRS**

```
<element name="_CRS" type="gml:AbstractCRSType" abstract="true" substitutionGroup="gml:_ReferenceSystem"/>

<complexType name="AbstractCRSType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractReferenceSystemType"/>
  </complexContent>
</complexType>
```

Abstract coordinate reference system, usually defined by a coordinate system and a datum. This abstract complexType shall not be used, extended, or restricted, in an Application Schema, to define a concrete subtype with a meaning equivalent to a concrete subtype specified in this document.

#### 12.2.4.6 **gml:crsRef**

```
<element name="crsRef" type="gml:CRSRefType" substitutionGroup="gml:referenceSystemRef"/>

<complexType name="CRSRefType">
  <complexContent>
    <restriction base="gml:ReferenceSystemRefType">
      <sequence>
        <element ref="gml:_CRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a CRS abstract coordinate reference system, either referencing or containing the definition of that CRS.

### 12.3 coordinateReferenceSystems schema

#### 12.3.1 Overview

##### 12.3.1.1 Introduction

A coordinate reference system consists of a coordinate system that is (usually) related to the earth through one datum. A coordinate reference system may be single, compound, or derived.

The coordinate system is composed of a set of coordinate axes with specified units of measure. This concept implies the mathematical rules that define how coordinate values are calculated from distances, angles and other geometric elements and vice versa.

The datum defines the origin of the coordinate system and ties it to the earth, ensuring that the abstract mathematical concept “coordinate system” can be applied to the practical problem of describing positions of features on or near the earth’s surface by means of coordinates. The datum implicitly (occasionally explicitly) contains the values chosen for the set parameters that represent the degrees of freedom of the coordinate system, as described in Subclause 12.1.2 above. Each datum subtype can be associated with only specific types of coordinate systems.

### 12.3.1.2 Principal types of coordinate reference systems

Geodetic survey practice usually divides coordinate reference systems into a number of sub-types. The common classification criterion for sub-typing of coordinate reference systems can be described as the way in which they deal with earth curvature. This has a direct effect on the portion of the earth’s surface that can be covered by that type of CRS with an acceptable degree of error.

Thus the following principal sub-types of coordinate reference system are distinguished:

- a) **Geocentric.** Type of coordinate reference system that deals with the earth’s curvature by taking the 3D spatial view, which obviates the need to model the earth’s curvature. The origin of a geocentric CRS is at the approximate centre of mass of the earth.
- b) **Geographic.** Type of coordinate reference system based on an ellipsoidal approximation of the geoid. This provides an accurate representation of the geometry of geographic features for a large portion of the earth’s surface. Geographic coordinate reference systems can be 2D or 3D. A 2D Geographic CRS is used when positions of features are described on the surface of the reference ellipsoid; a 3D Geographic CRS is used when positions are described on, above or below the reference ellipsoid.
- c) **Projected.** Type of coordinate reference system that is based on an approximation of the shape of the earth’s surface by a plane. The distortion that is inherent in the approximation is carefully controlled and known. Distortion correction is commonly applied to calculated bearings and distances to produce values that are a close match to actual field values.
- d) **Engineering.** Type of coordinate reference system that is used only in a contextually local sense. This sub-type is used to model two broad categories of local coordinate reference systems:
  - earth-fixed systems, applied to engineering activities on or near the surface of the earth;
  - spatial referencing by coordinates on moving platforms such as road vehicles, vessels, aircraft or spacecraft.

More information on Engineering CRSs is provided in Subclause 12.1.4.

- e) **Image.** An Image CRS is an Engineering CRS applied to images. Image CRSs warrant treatment as a separate sub-type because a separate user community exists for images with its own vocabulary. The definition of the associated Image Datum contains two data attributes not relevant for other datums and coordinate reference systems.
- f) **Vertical.** Type of coordinate reference system used for the recording of heights or depths. Vertical CRSs make use of the direction of gravity to define the concept of height or depth, but its relationship with gravity may not be straightforward. By implication ellipsoidal heights ( $h$ ) cannot be captured in a vertical coordinate reference system. Ellipsoidal heights cannot exist independently, but only as inseparable part of a 3D coordinate tuple defined in a geographic 3D coordinate reference system.
- g) **Temporal.** Used for the recording of time in association with any of the listed spatial coordinate reference systems only.

**EDITOR'S NOTE** The CRS WG of the OGC has agreed to expand the types of CRS described above to also allow defining Earth Centered Inertial CRSs, which do not rotate with the Earth. However, the changes needed to do this have not yet been determined.

#### 12.3.1.3 Compound coordinate reference systems

The traditional separation of horizontal and vertical position has resulted in different coordinate reference systems that are horizontal (2D) in nature and vertical (1D). Established practice combines the horizontal coordinates of a point with a height or depth from a different coordinate reference system to construct valid coordinates in 3D space. The coordinate reference system to which these 3D coordinates are referenced combines the separate horizontal and vertical coordinate reference systems of the horizontal and vertical coordinates. Such a coordinate system is called a compound coordinate reference system.

A Compound CRS is thus a coordinate reference system that combines an ordered sequence of two or more coordinate reference systems, none of which can themselves be compound. In general, a Compound CRS may contain any number of axes. The Compound CRS contains an ordered set of coordinate reference systems and the tuple order of a compound coordinate set shall follow that order, while the subsets of the tuple, described by each of the composing coordinate reference systems, shall follow the tuple order valid for their respective coordinate reference systems.

For spatial coordinates, a number of constraints exist for the construction of compound CRSs. For example, the coordinate reference systems that are combined should not contain any duplicate or redundant axes. Valid combinations include:

- a) Geographic 2D + Vertical
- b) Geographic 2D + Engineering 1D (near vertical)
- c) Projected + Vertical
- d) Projected + Engineering 1D (near vertical)
- e) Engineering (horizontal 2D or 1D linear) + Vertical

Any coordinate reference system or any of the above listed combinations of coordinate reference systems can have a temporal coordinate reference system added. More than one temporal coordinate reference system may be added if these axes represent different time quantities. For example, the oil industry sometimes uses "4D seismic", by which is meant seismic data with the vertical axis expressed in milliseconds (signal travel time). A second time axis indicates how it changes with time (years), e.g. as a reservoir is gradually exhausted of its recoverable oil or gas).

#### 12.3.1.4 Derived coordinate reference systems

Some coordinate reference systems are defined by applying a coordinate conversion to another coordinate reference system. Such a coordinate reference system is called a Derived CRS, and the coordinate reference system it was derived from by applying the conversion is called the Source or Base CRS. A coordinate conversion is an arithmetic operation with zero or more parameters that have defined values. The Source CRS and Derived CRS have the same Datum. The best-known example of a Derived CRS is a Projected CRS, which is always derived from a source Geographic CRS by applying the coordinate conversion known as a map projection.

In principle, all sub-types of coordinate reference system may take on the role of either Source or Derived CRS with the exception of a Geocentric CRS and a Projected CRS. The latter is modelled as an object class under its own name, rather than as a general Derived CRS of type "projected". This has been done to honour common practice, which acknowledges Projected CRS'sCRSs as one of the best known types of coordinate reference systems.

An example of a Derived CRS of *derivedCRSType*: “geographic” is one of which the unit of measure has been modified with respect to an earlier defined Geographic CRS, which then takes the role of Source CRS.

#### 12.3.1.5 Schema introduction

The spatial-temporal coordinate reference systems schema, coordinateReferenceSystems.xsd, has two logical parts. One part defines elements and types for XML encoding of abstract coordinate reference systems definitions. The larger part defines specialized constructs for XML encoding of definitions of the multiple concrete types of spatial-temporal coordinate reference systems.

This schema encodes the Coordinate Reference System package of the UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates, with the exception of the abstract "SC\_CRS" class. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex B of Topic 2.

The coordinateReferenceSystems schema includes the coordinateSystems.xsd, datums.xsd, and coordinateOperations.xsd GML schemas. This schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd: coordinateReferenceSystems:v3.1.0

#### 12.3.2 Abstract coordinate reference systems

##### 12.3.2.1 gml:\_CoordinateReferenceSystem

```
<element name="_CoordinateReferenceSystem" type="gml:AbstractCoordinateReferenceSystemType"  
abstract="true" substitutionGroup="gml:_CRS"/>  
  
<complexType name="AbstractCoordinateReferenceSystemType" abstract="true">  
    <complexContent>  
        <extension base="gml:AbstractCRSType"/>  
    </complexContent>  
</complexType>
```

A coordinate reference system consists of an ordered sequence of coordinate system axes that are related to the earth through a datum. A coordinate reference system is defined by one datum and by one coordinate system. Most coordinate reference system do not move relative to the earth, except for engineering coordinate reference systems defined on moving platforms such as cars, ships, aircraft, and spacecraft. For further information, see Subclause 12.3.1.

Coordinate reference systems are commonly divided into sub-types. The common classification criterion for sub-typing of coordinate reference systems is the way in which they deal with earth curvature. This has a direct effect on the portion of the earth's surface that can be covered by that type of CRS with an acceptable degree of error. The exception to the rule is the subtype "Temporal" which has been added by analogy.

##### 12.3.2.2 gml:coordinateReferenceSystemRef

```
<element name="coordinateReferenceSystemRef" type="gml:CoordinateReferenceSystemRefType"  
substitutionGroup="gml:crsRef"/>  
  
<complexType name="CoordinateReferenceSystemRefType">  
    <complexContent>  
        <restriction base="gml:CRSRefType">  
            <sequence>  
                <element ref="gml:_CoordinateReferenceSystem" minOccurs="0"/>  
            </sequence>  
            <attributeGroup ref="gml:AssociationAttributeGroup"/>  
        </restriction>  
    </complexContent>  
</complexType>
```

Association to a coordinate reference system, either referencing or containing the definition of that reference system.

#### 12.3.2.3 **gml:\_GeneralDerivedCRS**

```
<element name="_GeneralDerivedCRS" type="gml:AbstractGeneralDerivedCRSType" abstract="true"
substitutionGroup="gml:_CoordinateReferenceSystem"/>

<complexType name="AbstractGeneralDerivedCRSType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <element ref="gml:baseCRS"/>
        <element ref="gml:definedByConversion"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A coordinate reference system that is defined by its coordinate conversion from another coordinate reference system (not by a datum). This abstract complexType shall not be used, extended, or restricted, in an Application Schema, to define a concrete subtype with a meaning equivalent to a concrete subtype specified in this document.

#### 12.3.2.4 **gml:baseCRS**

```
<element name="baseCRS" type="gml:CoordinateReferenceSystemRefType"/>
```

Association to the coordinate reference system used by this derived CRS.

#### 12.3.2.5 **gml:definedByConversion**

```
<element name="definedByConversion" type="gml:GeneralConversionRefType"/>
```

Association to the coordinate conversion used to define this derived CRS.

#### 12.3.2.6 **gml:generalDerivedCRSRef**

```
<element name="generalDerivedCRSRef" type="gml:GeneralDerivedCRSRefType"
substitutionGroup="gml:coordinateReferenceSystemRef"/>

<complexType name="GeneralDerivedCRSRefType">
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:_GeneralDerivedCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a general derived coordinate reference system, either referencing or containing the definition of that reference system.

### 12.3.3 Concrete coordinate reference systems

#### 12.3.3.1 **gml:CompoundCRS**

```
<element name="CompoundCRS" type="gml:CompoundCRSType" substitutionGroup="gml:_CRS"/>
```

```

<complexType name="CompoundCRSType">
  <complexContent>
    <extension base="gml:AbstractCRSType">
      <sequence>
        <element ref="gml:includesCRS" minOccurs="2" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

A coordinate reference system describing the position of points through two or more independent coordinate reference systems.

The includesCRS elements are an ordered sequence of associations to all the component coordinate reference systems included in this compound coordinate reference system.

#### 12.3.3.2 gml:includesCRS

```
<element name="includesCRS" type="gml:CoordinateReferenceSystemRefType"/>
```

An association to a component coordinate reference system included in this compound coordinate reference system.

#### 12.3.3.3 gml:compoundCRSRef

```

<element name="compoundCRSRef" type="gml:CompoundCRSRefType" substitutionGroup="gml:crsRef"/>

<complexType name="CompoundCRSRefType">
  <complexContent>
    <restriction base="gml:CRSRefType">
      <sequence>
        <element ref="gml:CompoundCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a compound coordinate reference system, either referencing or containing the definition of that reference system.

#### 12.3.3.4 gml:GeographicCRS

```

<element name="GeographicCRS" type="gml:GeographicCRSType"
  substitutionGroup="gml:_CoordinateReferenceSystem"/>

<complexType name="GeographicCRSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <element ref="gml:usesEllipsoidalCS"/>
        <element ref="gml:usesGeodeticDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

A coordinate reference system based on an ellipsoidal approximation of the geoid; this provides an accurate representation of the geometry of geographic features for a large portion of the earth's surface.

### 12.3.3.5 **gml:usesEllipsoidalCS**

```
<element name="usesEllipsoidalCS" type="gml:EllipsoidalCSRefType"/>
```

Association to the ellipsoidal coordinate system used by this CRS.

### 12.3.3.6 **gml:usesGeodeticDatum**

```
<element name="usesGeodeticDatum" type="gml:GeodeticDatumRefType"/>
```

Association to the geodetic datum used by this CRS

### 12.3.3.7 **gml:geographicCRSRef**

```
<element name="geographicCRSRef" type="gml:GeographicCRSRefType"
substitutionGroup="gml:coordinateReferenceSystemRef">

<complexType name="GeographicCRSRefType">
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:GeographicCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a geographic coordinate reference system, either referencing or containing the definition of that reference system.

### 12.3.3.8 **gml:VerticalCRS**

```
<element name="VerticalCRS" type="gml:VerticalCRSType"
substitutionGroup="gml:_CoordinateReferenceSystem">

<complexType name="VerticalCRSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <element ref="gml:usesVerticalCS"/>
        <element ref="gml:usesVerticalDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A 1D coordinate reference system used for recording heights or depths. Vertical CRSSs make use of the direction of gravity to define the concept of height or depth, but the relationship with gravity may not be straightforward. By implication, ellipsoidal heights ( $h$ ) cannot be captured in a vertical coordinate reference system. Ellipsoidal heights cannot exist independently, but only as an inseparable part of a 3D coordinate tuple defined in a geographic 3D coordinate reference system.

### 12.3.3.9 **gml:usesVerticalCS**

```
<element name="usesVerticalCS" type="gml:VerticalCSRefType"/>
```

Association to the vertical coordinate system used by this CRS.

### 12.3.3.10 **gml:usesVerticalDatum**

```
<element name="usesVerticalDatum" type="gml:VerticalDatumRefType"/>
```

Association to the vertical datum used by this CRS.

#### 12.3.3.11 **gml:verticalCRSRef**

```
<element name="verticalCRSRef" type="gml:VerticalCRSRefType"
substitutionGroup="gml:coordinateReferenceSystemRef"/>

<complexType name="VerticalCRSRefType">
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:VerticalCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a vertical coordinate reference system, either referencing or containing the definition of that reference system.

#### 12.3.3.12 **gml:GeocentricCRS**

```
<element name="GeocentricCRS" type="gml:GeocentricCRSType"
substitutionGroup="gml:_CoordinateReferenceSystem"/>

<complexType name="GeocentricCRSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <choice>
          <element ref="gml:usesCartesianCS"/>
          <element ref="gml:usesSphericalCS"/>
        </choice>
        <element ref="gml:usesGeodeticDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A 3D coordinate reference system with the origin at the approximate centre of mass of the earth. A geocentric CRS deals with the earth's curvature by taking a 3D spatial view, which obviates the need to model the earth's curvature.

#### 12.3.3.13 **gml:usesCartesianCS**

```
<element name="usesCartesianCS" type="gml:CartesianCSRefType"/>
```

Association to the Cartesian coordinate system used by this CRS.

#### 12.3.3.14 **gml:usesSphericalCS**

```
<element name="usesSphericalCS" type="gml:SphericalCSRefType"/>
```

Association to the spherical coordinate system used by this CRS.

#### 12.3.3.15 **gml:geocentricCRSRef**

```
<element name="geocentricCRSRef" type="gml:GeocentricCRSRefType"
substitutionGroup="gml:coordinateReferenceSystemRef"/>
```

```

<complexType name="GeocentricCRSRefType">
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:GeocentricCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a geocentric coordinate reference system, either referencing or containing the definition of that reference system.

#### 12.3.3.16 gml:ProjectedCRS

```

<element name="ProjectedCRS" type="gml:ProjectedCRSType" substitutionGroup="gml:_GeneralDerivedCRS"/>

<complexType name="ProjectedCRSType">
  <complexContent>
    <extension base="gml:AbstractGeneralDerivedCRSType">
      <sequence>
        <element ref="gml:usesCartesianCS"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

A 2D coordinate reference system used to approximate the shape of the earth on a planar surface, but in such a way that the distortion that is inherent to the approximation is carefully controlled and known. Distortion correction is commonly applied to calculated bearings and distances to produce values that are a close match to actual field values.

#### 12.3.3.17 gml:projectedCRSRef

```

<element name="projectedCRSRef" type="gml:ProjectedCRSRefType"
  substitutionGroup="gml:generalDerivedCRSRef"/>

<complexType name="ProjectedCRSRefType">
  <complexContent>
    <restriction base="gml:GeneralDerivedCRSRefType">
      <sequence>
        <element ref="gml:ProjectedCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a projected coordinate reference system, either referencing or containing the definition of that reference system.

#### 12.3.3.18 gml:DerivedCRS

```

<element name="DerivedCRS" type="gml:DerivedCRSType" substitutionGroup="gml:_GeneralDerivedCRS"/>

<complexType name="DerivedCRSType">
  <complexContent>
    <extension base="gml:AbstractGeneralDerivedCRSType">
      <sequence>
        <element ref="gml:derivedCRSType"/>
        <element ref="gml:usesCS"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

        </extension>
    </complexContent>
</complexType>
```

A coordinate reference system that is defined by its coordinate conversion from another coordinate reference system but is not a projected coordinate reference system. This category includes coordinate reference systems derived from a projected coordinate reference system.

#### 12.3.3.19 **gml:derivedCRSType**

```

<element name="derivedCRSType" type="gml:DerivedCRSTypeType"/>

<complexType name="DerivedCRSTypeType">
    <simpleContent>
        <restriction base="gml:CodeType">
            <attribute name="codeSpace" type="anyURI" use="required"/>
        </restriction>
    </simpleContent>
</complexType>
```

Type of a derived coordinate reference system. The required codeSpace attribute shall reference a source of information specifying the values and meanings of all the allowed string values for this DerivedCRSTypeType.

#### 12.3.3.20 **gml:usesCS**

```
<element name="usesCS" type="gml:CoordinateSystemRefType"/>
```

Association to the coordinate system used by this CRS.

#### 12.3.3.21 **gml:derivedCRSRef**

```

<element name="derivedCRSRef" type="gml:DerivedCRSRefType"
substitutionGroup="gml:generalDerivedCRSRef"/>

<complexType name="DerivedCRSRefType">
    <complexContent>
        <restriction base="gml:GeneralDerivedCRSRefType">
            <sequence>
                <element ref="gml:DerivedCRS" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
```

Association to a non-projected derived coordinate reference system, either referencing or containing the definition of that reference system.

#### 12.3.3.22 **gml:EngineeringCRS**

```

<element name="EngineeringCRS" type="gml:EngineeringCRSType"
substitutionGroup="gml:_CoordinateReferenceSystem"/>

<complexType name="EngineeringCRSType">
    <complexContent>
        <extension base="gml:AbstractCoordinateReferenceSystemType">
            <sequence>
                <element ref="gml:usesCS"/>
                <element ref="gml:usesEngineeringDatum"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

```
</complexType>
```

A contextually local coordinate reference system; which can be divided into two broad categories:

- a) earth-fixed systems applied to engineering activities on or near the surface of the earth;
- b) CRSs on moving platforms such as road vehicles, vessels, aircraft, or spacecraft.

For further information, see Subclause 12.1.4.

#### **12.3.3.23 `gml:usesEngineeringDatum`**

```
<element name="usesEngineeringDatum" type="gml:EngineeringDatumRefType"/>
```

Association to the engineering datum used by this CRS.

#### **12.3.3.24 `gml:engineeringCRSRef`**

```
<element name="engineeringCRSRef" type="gml:EngineeringCRSRefType"
substitutionGroup="gml:coordinateReferenceSystemRef"/>

<complexType name="EngineeringCRSRefType">
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:EngineeringCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to an engineering coordinate reference system, either referencing or containing the definition of that reference system.

#### **12.3.3.25 `gml:ImageCRS`**

```
<element name=" " type="gml:ImageCRSType" substitutionGroup="gml:_CoordinateReferenceSystem"/>

<complexType name="ImageCRSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <choice>
          <element ref="gml:usesCartesianCS"/>
          <element ref="gml:usesObliqueCartesianCS"/>
        </choice>
        <element ref="gml:usesImageDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

An engineering coordinate reference system applied to locations in images. Image coordinate reference systems are treated as a separate sub-type because a separate user community exists for images with its own terms of reference.

#### **12.3.3.26 `gml:usesObliqueCartesianCS`**

```
<element name="usesObliqueCartesianCS" type="gml:ObliqueCartesianCSRefType"/>
```

Association to the oblique Cartesian coordinate system used by this CRS.

### **12.3.3.27 `gml:usesImageDatum`**

```
<element name="usesImageDatum" type="gml:ImageDatumRefType"/>
```

Association to the image datum used by this CRS.

### **12.3.3.28 `gml:imageCRSRef`**

```
<element name=" " type="gml:ImageCRSRefType" substitutionGroup="gml:coordinateReferenceSystemRef"/>

<complexType name="ImageCRSRefType">
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:ImageCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to an image coordinate reference system, either referencing or containing the definition of that reference system.

### **12.3.3.29 `gml:TemporalCRS`**

```
<element name="TemporalCRS" type="gml:TemporalCRSType"
  substitutionGroup="gml:_CoordinateReferenceSystem"/>

<complexType name="TemporalCRSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <element ref="gml:usesTemporalCS"/>
        <element ref="gml:usesTemporalDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A 1D coordinate reference system used for the recording of time.

### **12.3.3.30 `gml:usesTemporalCS`**

```
<element name="usesTemporalCS" type="gml:TemporalCSRefType"/>
```

Association to the temporal coordinate system used by this CRS.

### **12.3.3.31 `gml:usesTemporalDatum`**

```
<element name="usesTemporalDatum" type="gml:TemporalDatumRefType"/>
```

Association to the temporal datum used by this CRS.

### **12.3.3.32 `gml:temporalCRSRef`**

```
<element name="temporalCRSRef" type="gml:TemporalCRSRefType"
  substitutionGroup="gml:coordinateReferenceSystemRef"/>

<complexType name="TemporalCRSRefType">
  <complexContent>
```

```

<restriction base="gml:CoordinateReferenceSystemRefType">
  <sequence>
    <element ref="gml:TemporalCRS" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</restriction>
</complexContent>
</complexType>

```

Association to a temporal coordinate reference system, either referencing or containing the definition of that reference system.

## 12.4 coordinateSystems schema

### 12.4.1 Overview

#### 12.4.1.1 Introduction

The coordinates of points are recorded in a coordinate system. A coordinate system is the set of coordinate system axes that spans the coordinate space. This concept implies the set of mathematical rules that determine how coordinates are associated with invariant quantities such as angles and distances. In other words, a coordinate system implies how coordinates are calculated from geometric elements such as distances and angles and vice versa. The calculus required to derive angles and distances from point coordinates and vice versa in a map plane is simple Euclidean 2D arithmetic. To do the same on the surface of an ellipsoid (curved 2D space) involves more complex ellipsoidal calculus. These rules cannot be specified in detail, but are implied in the geometric properties of the coordinate space.

NOTE: The word ‘distances’ is used loosely in the above description. Strictly speaking distances are not invariant quantities, as they are expressed in the unit of measure defined for the coordinate system; ratios of distances are invariant.

One coordinate system may be used by multiple coordinate reference systems. A coordinate system is composed of an ordered sequence of axes, the number of axes being equal to the dimension of the space of which it describes the geometry. Coordinates in coordinate tuples shall be listed in the specified axes sequence and units.

The dimension of the coordinate space, the names, the units of measure, the directions and sequence of the axes are all part of the Coordinate System definition. The number of coordinates in a tuple and the number of axes in a coordinate system shall be equal.

Example: It is therefore not permitted to supply a coordinate tuple with two heights of different definition in the same coordinate tuple.

#### 12.4.1.2 Coordinate system types

Coordinate systems are divided in types by the geometric properties of the coordinate space spanned and the geometric properties of the axes themselves (straight or curved; perpendicular or not). Certain types of coordinate system can only be used with specific subtypes of coordinate reference system. A description of coordinate system types is provided in Table TBD.

**Table 3 — Types of coordinate systems with CRS constraints**

CS type	Description	Used by CRS type
Cartesian	1-, 2-, or 3-dimensional coordinate system that gives the position of points relative to 2 or 3 orthogonal straight axes in the 2- or 3-dimensional case. In the 1-dimensional case, it contains a single straight coordinate axis. In the multi-dimensional case all axes shall have the same unit of measure.	Geocentric Projected Engineering Image Derived

oblique Cartesian	2- or 3-dimensional coordinate system with straight axes that are not necessarily orthogonal	Engineering Image Derived
ellipsoidal	2- or 3-dimensional coordinate system in which position is specified by geodetic latitude, geodetic longitude and (in the three-dimensional case) ellipsoidal height, associated with one or more geographic coordinate reference systems.	Geographic Engineering Derived
vertical	1-dimensional coordinate system used to record the heights (or depths) of points dependent on the Earth's gravity field. An exact definition is deliberately not provided as the complexities of the subject fall outside the scope of this specification.	Vertical Engineering Derived
linear	1-dimensional coordinate system that consists of the points which lie on the single axis described. The associated ordinate is the distance from the specified origin to the point along the axis. Example: usage of the line feature representing a road to describe points on or along that road.	Engineering Derived
user-defined	2- or 3-dimensional coordinate system that consists of any combination of coordinate axes not covered by any other coordinate system type. An example is a multilinear coordinate system which contains one coordinate axis that may have any 1-D shape which has no intersections with itself. This non-straight axis is supplemented by one or two straight axes to complete a 2 or 3 dimensional coordinate system.	Engineering Derived
temporal	1-dimensional coordinate system containing a single time axis and used to describe the temporal position of a point in the specified time units from a specified time origin.	Temporal
spherical	3-dimensional coordinate system with one distance, measured from the origin, and two angular coordinates; not to be confused with an ellipsoidal coordinate system based on an ellipsoid 'degenerated' into a sphere	Geocentric Engineering Derived
cylindrical	3-dimensional coordinate system consisting of a polar coordinate system extended by a straight coordinate axis perpendicular to the plane spanned by the polar coordinate system.	Engineering Derived
polar	2-dimensional coordinate system in which position is specified by the distance from the origin and the angle between the line from origin to point and a reference direction.	Engineering Derived

#### 12.4.1.3 Coordinate system axis

A coordinate system is composed of an ordered set of coordinate system axes. Each of its axes is completely characterised by a unique combination of axis name, axis abbreviation, axis direction and axis unit of measure.

The concept of coordinate axis requires some clarification. Consider an arbitrary  $x$ ,  $y$ ,  $z$  coordinate system. The  $x$ -axis may be defined as the locus of points with  $y = z = 0$ . This is easily enough understood if the  $x$ ,  $y$ ,  $z$  coordinate system is a Cartesian system and the space it describes is Euclidean. It becomes a bit more difficult to understand in the case of a strongly curved space, such as the surface of an ellipsoid, its geometry described by an ellipsoidal coordinate system (2D or 3D). Applying the same definition by analogy to the curvilinear *latitude* and *longitude* coordinates, the latitude axis would be the equator and the longitude axis would be the prime meridian, which is not a satisfactory definition.

Bearing in mind that the order of the coordinates in a coordinate tuple shall be the same as the defined order of the coordinate axes, the ' $i$ 'th coordinate axis of a coordinate system is defined as the locus of points for which all coordinates with sequence number not equal to ' $i$ ', have a constant value locally (whereby  $i = 1 \dots n$ , and  $n$  is the dimension of the coordinate space).

It will be evident that the addition of the word 'locally' in this definition apparently adds an element of ambiguity and this is intentional. However, the definition of the coordinate parameter associated with any axis shall be unique. The coordinate axis itself should not be interpreted as a unique mathematical object, the associated coordinate parameter should.

**Example 1** Geodetic latitude is defined as the “Angle from the equatorial plane to the perpendicular to the ellipsoid through a given point, northwards usually treated as positive”. However, when used in an ellipsoidal coordinate system, the geodetic latitude axis will be described as pointing ‘north’. In two different points on the ellipsoid the direction ‘north’ will be a spatially different direction, but the concept of latitude is the same.

Furthermore the specified directions of the coordinate axes are often only approximate; two geographic coordinate reference systems will make use of the same ellipsoidal coordinate system. These coordinate systems are associated with the earth through two different geodetic datums, which may lead to the two systems being slightly rotated w.r.t. each other.

Usage of coordinate system axis names is constrained by geodetic custom in a number of cases, depending mainly on the coordinate reference system type. These constraints are shown in Table TBD. These constraints work in two directions; for example the names ‘geodetic latitude’ and ‘geodetic longitude’ shall be used to designate the coordinate axis names associated with a geographic coordinate reference system. Conversely, these names shall not be used in any other context. Image and engineering coordinate reference systems may make use of names specific to the local context or custom and are therefore not included as constraints in this list.

**Table 4 — Naming constraints for coordinate system axes**

CS	CRS	Permitted coordinate system axis names
Cartesian	Geocentric	Geocentric X, Geocentric Y, Geocentric Z
Spherical	Geocentric	Spherical Latitude, Spherical Longitude, Geocentric Radius
Ellipsoidal	Geographic	Geodetic latitude, Geodetic longitude, Ellipsoidal height (if 3D)
Vertical	Vertical	Gravity-related height
Vertical	Vertical	Depth
Cartesian	Projected	Easting, Northing
Cartesian	Projected	Westing, Southing

**Example 2** The combination {Latitude, Lat, north, degree} would lead to one instance of the object class ‘coordinate system axis’; the combination {Latitude, φ, north, degree} to another instance, the axis abbreviation being different.

#### 12.4.1.4 Schema introduction

The coordinate systems schema, coordinateSystems.xsd, has three logical parts, which define elements and types for XML encoding of the definitions of:

- a) Coordinate system axes
- b) Abstract coordinate system
- c) Multiple concrete types of spatial-temporal coordinate systems

This schema encodes the Coordinate System package of the UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex B of Topic 2.

The referenceSystems schema includes the referenceSystems.xsd GML schema. This schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd: coordinateSystems:v3.1.0

## 12.4.2 Coordinate system axes

### 12.4.2.1 **gml:CoordinateSystemAxis**

```
<element name="CoordinateSystemAxis" type="gml:CoordinateSystemAxisType"
substitutionGroup="gml:Definition"/>

<complexType name="CoordinateSystemAxisType">
  <complexContent>
    <extension base="gml:CoordinateSystemAxis BaseType">
      <sequence>
        <element ref="gml:axisID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:axisAbbrev"/>
        <element ref="gml:axisDirection"/>
      </sequence>
      <attribute ref="gml:uom" use="required"/>
    </extension>
  </complexContent>
</complexType>

<complexType name="CoordinateSystemAxis BaseType" abstract="true">
  <complexContent>
    <restriction base="gml:DefinitionType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:axisName"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
```

Definition of a coordinate system axis.

The CoordinateSystemAxis BaseType provides the basic encoding for coordinate system axis objects, simplifying and restricting the DefinitionType as needed.

The "axisID" elements contain a set of alternative identifications of this coordinate system axis. The first axisID, if any, is normally the primary identification code, and any others are aliases.

The "remarks" element contains comments on or information about this reference system, including source information.

### 12.4.2.2 **gml:axisName**

```
<element name="axisName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which this coordinate system axis is identified.

### 12.4.2.3 **gml:axisID**

```
<element name="axisID" type="gml:IdentifierType"/>
```

An identification of a coordinate system axis.

### 12.4.2.4 **gml:axisAbbrev**

```
<element name="axisAbbrev" type="gml:CodeType"/>
```

The abbreviation used for this coordinate system axis. This abbreviation can be used to identify the ordinates in a coordinate tuple. Examples are X and Y. The codeSpace attribute can reference a source of more information on a set of standardized abbreviations, or on this abbreviation.

#### 12.4.2.5 **gml:axisDirection**

```
<element name="axisDirection" type="gml:CodeType"/>
```

Direction of this coordinate system axis (or in the case of Cartesian projected coordinates, the direction of this coordinate system axis at the origin). Examples: north or south, east or west, up or down. Within any set of coordinate system axes, only one of each pair of terms can be used. For earth-fixed CRSs, this direction is often approximate and intended to provide a human interpretable meaning to the axis. When a geodetic datum is used, the precise directions of the axes may therefore vary slightly from this approximate direction. Note that an EngineeringCRS can include specific descriptions of the directions of its coordinate system axes. For example, the path of a linear CRS axis can be referenced in another document, such as referencing a GML feature that references or includes a curve geometry. The codeSpace attribute can reference a source of more information on a set of standardized directions, or on this direction.

#### 12.4.2.6 **gml:uom**

```
<attribute name="uom" type="anyURI"/>
```

Identifier of the unit of measure used for this coordinate system axis. The value of this coordinate in a coordinate tuple shall be recorded using this unit of measure, whenever those coordinates use a coordinate reference system that uses a coordinate system that uses this axis.

#### 12.4.2.7 **gml:coordinateSystemAxisRef**

```
<element name="coordinateSystemAxisRef" type="gml:CoordinateSystemAxisRefType"
substitutionGroup="gml:dictionaryEntry"/>

<complexType name="CoordinateSystemAxisRefType">
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:CoordinateSystemAxis" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a coordinate system axis, either referencing or containing the definition of that axis.

### 12.4.3 Abstract coordinate system

#### 12.4.3.1 **gml:\_CoordinateSystem**

```
<element name="_CoordinateSystem" type="gml:AbstractCoordinateSystemType" abstract="true"
substitutionGroup="gml:Definition"/>

<complexType name="AbstractCoordinateSystemType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemBaseType">
      <sequence>
        <element ref="gml:csID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:usesAxis" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

<complexType name="AbstractCoordinateSystemBaseType" abstract="true">
  <complexContent>
    <restriction base="gml:DefinitionType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:csName"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>

```

A coordinate system (CS) is the set of coordinate system axes that spans a given coordinate space. A CS is derived from a set of (mathematical) rules for specifying how coordinates in a given space are to be assigned to points. This abstract complexType shall not be used, extended, or restricted, in an Application Schema, to define a concrete subtype with a meaning equivalent to a concrete subtype specified in this document.

The AbstractCoordinateSystemBaseType provides the basic encoding for coordinate system objects, simplifying and restricting the DefinitionType as needed.

The "csID" elements contain a set of alternative identifications of this coordinate system. The first csID, if any, is normally the primary identification code, and any others are aliases.

The usesAxis elements are an ordered sequence of associations to the coordinate system axes included in this coordinate system. The coordinate values in a coordinate tuple shall be recorded in the order in which the coordinate system axes associations are recorded, whenever those coordinates use a coordinate reference system that uses this coordinate system.

The "remarks" element contains comments on or information about this reference system, including source information.

#### **12.4.3.2 gml:csName**

```
<element name="csName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which this coordinate system is identified.

#### **12.4.3.3 gml:csID**

```
<element name="csID" type="gml:IdentifierType"/>
```

An identification of a coordinate system.

#### **12.4.3.4 gml:usesAxis**

```
<element name="usesAxis" type="gml:CoordinateSystemAxisRefType" substitutionGroup="gml:dictionaryEntry"/>
```

#### **12.4.3.5 Association to a coordinate system axis.gml:coordinateSystemRef**

```

<element name="coordinateSystemRef" type="gml:CoordinateSystemRefType"/>

<complexType name="CoordinateSystemRefType">
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:_CoordinateSystem" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

```

        </restriction>
    </complexContent>
</complexType>
```

Association to a coordinate system, either referencing or containing the definition of that coordinate system.

#### 12.4.4 Concrete coordinate systems

##### 12.4.4.1 gml:EllipsoidalCS

```

<element name="EllipsoidalCS" type="gml:EllipsoidalCSType" substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="EllipsoidalCSType">
    <complexContent>
        <extension base="gml:AbstractCoordinateSystemType"/>
    </complexContent>
</complexType>
```

A two- or three-dimensional coordinate system in which position is specified by geodetic latitude, geodetic longitude, and (in the three-dimensional case) ellipsoidal height. An EllipsoidalCS shall have two or three usesAxis associations.

##### 12.4.4.2 gml:ellipsoidalCSRef

```

<element name="ellipsoidalCSRef" type="gml:EllipsoidalCSRefType"
substitutionGroup="gml:coordinateSystemRef"/>

<complexType name="EllipsoidalCSRefType">
    <complexContent>
        <restriction base="gml:CoordinateSystemRefType">
            <sequence>
                <element ref="gml:EllipsoidalCS" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
```

Association to an ellipsoidal coordinate system, either referencing or containing the definition of that coordinate system.

##### 12.4.4.3 gml:CartesianCS

```

<element name="CartesianCS" type="gml:CartesianCSType" substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="CartesianCSType">
    <complexContent>
        <extension base="gml:AbstractCoordinateSystemType"/>
    </complexContent>
</complexType>
```

A 1-, 2-, or 3-dimensional coordinate system. Gives the position of points relative to orthogonal straight axes in the 2- and 3-dimensional cases. In the 1-dimensional case, it contains a single straight coordinate axis. In the multi-dimensional case, all axes shall have the same length unit of measure. A CartesianCS shall have one, two, or three usesAxis associations.

##### 12.4.4.4 gml:cartesianCSRef

```

<element name="cartesianCSRef" type="gml:CartesianCSRefType" substitutionGroup="gml:coordinateSystemRef"/>

<complexType name="CartesianCSRefType">
    <complexContent>
```

```

<restriction base="gml:CoordinateSystemRefType">
  <sequence>
    <element ref="gml:CartesianCS" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</restriction>
</complexContent>
</complexType>

```

Association to a Cartesian coordinate system, either referencing or containing the definition of that coordinate system.

#### 12.4.4.5 gml:VerticalCS

```

<element name="VerticalCS" type="gml:VerticalCSType" substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="VerticalCSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>

```

A one-dimensional coordinate system used to record the heights (or depths) of points. Such a coordinate system is usually dependent on the Earth's gravity field, perhaps loosely as when atmospheric pressure is the basis for the vertical coordinate system axis. A VerticalCS shall have one usesAxis association.

#### 12.4.4.6 gml:verticalCSRef

```

<element name="verticalCSRef" type="gml:VerticalCSRefType" substitutionGroup="gml:coordinateSystemRef"/>

<complexType name="VerticalCSRefType">
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:VerticalCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a vertical coordinate system, either referencing or containing the definition of that coordinate system.

#### 12.4.4.7 gml:TemporalCS

```

<element name="TemporalCS" type="gml:TemporalCSType" substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="TemporalCSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>

```

A one-dimensional coordinate system containing a single time axis, used to describe the temporal position of a point in the specified time units from a specified time origin. A TemporalCS shall have one usesAxis association.

#### 12.4.4.8 gml:temporalCSRef

```

<element name="temporalCSRef" type="gml:TemporalCSRefType" substitutionGroup="gml:coordinateSystemRef"/>

```

```

<complexType name="TemporalCSRefType">
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:TemporalCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a temporal coordinate system, either referencing or containing the definition of that coordinate system.

#### 12.4.4.9 gml:LinearCS

```

<element name="LinearCS" type="gml:LinearCSType" substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="LinearCSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>

```

A one-dimensional coordinate system that consists of the points that lie on the single axis described. The associated ordinate is the distance from the specified origin to the point along the axis. Example: usage of the line feature representing a road to describe points on or along that road. A LinearCS shall have one usesAxis association.

#### 12.4.4.10 gml:linearCSRef

```

<element name="linearCSRef" type="gml:LinearCSRefType" substitutionGroup="gml:coordinateSystemRef"/>

<complexType name="LinearCSRefType">
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:LinearCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a linear coordinate system, either referencing or containing the definition of that coordinate system.

#### 12.4.4.11 gml:UserDefinedCS

```

<element name="UserDefinedCS" type="gml:UserDefinedCSType" substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="UserDefinedCSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>

```

A two- or three-dimensional coordinate system that consists of any combination of coordinate axes not covered by any other coordinate system type. An example is a multilinear coordinate system which contains one coordinate axis that may have any 1-D shape which has no intersections with itself. This non-straight axis is supplemented by one or two straight axes to complete a 2 or 3 dimensional coordinate system. The non-straight axis is typically incrementally straight or curved. A UserDefinedCS shall have two or three usesAxis associations.

#### **12.4.4.12 `gml:userDefinedCSRef`**

```
<element name="userDefinedCSRef" type="gml:UserDefinedCSRefType"
substitutionGroup="gml:coordinateSystemRef"/>

<complexType name="UserDefinedCSRefType">
<complexContent>
<restriction base="gml:CoordinateSystemRefType">
<sequence>
<element ref="gml:UserDefinedCS" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</restriction>
</complexContent>
</complexType>
```

Association to a user-defined coordinate system, either referencing or containing the definition of that coordinate system.

#### **12.4.4.13 `gml:SphericalCS`**

```
<element name="SphericalCS" type="gml:SphericalCSType" substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="SphericalCSType">
<complexContent>
<extension base="gml:AbstractCoordinateSystemType"/>
</complexContent>
</complexType>
```

A three-dimensional coordinate system with one distance measured from the origin and two angular coordinates. Not to be confused with an ellipsoidal coordinate system based on an ellipsoid "degenerated" into a sphere. A SphericalCS shall have three usesAxis associations.

#### **12.4.4.14 `gml:sphericalCSRef`**

```
<element name="sphericalCSRef" type="gml:SphericalCSRefType" substitutionGroup="gml:coordinateSystemRef"/>

<complexType name="SphericalCSRefType">
<complexContent>
<restriction base="gml:CoordinateSystemRefType">
<sequence>
<element ref="gml:SphericalCS" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</restriction>
</complexContent>
</complexType>
```

Association to a spherical coordinate system, either referencing or containing the definition of that coordinate system.

#### **12.4.4.15 `gml:PolarCS`**

```
<element name="PolarCS" type="gml:PolarCSType" substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="PolarCSType">
<complexContent>
<extension base="gml:AbstractCoordinateSystemType"/>
</complexContent>
</complexType>
```

A two-dimensional coordinate system in which position is specified by the distance from the origin and the angle between the line from the origin to a point and a reference direction. A PolarCS shall have two usesAxis associations.

#### 12.4.4.16 gml:polarCSRef

```
<element name="polarCSRef" type="gml:PolarCSRefType" substitutionGroup="gml:coordinateSystemRef"/>

<complexType name="PolarCSRefType">
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:PolarCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a polar coordinate system, either referencing or containing the definition of that coordinate system.

#### 12.4.4.17 gml:CylindricalCS

```
<element name="CylindricalCS" type="gml:CylindricalCSType" substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="CylindricalCSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
```

A three-dimensional coordinate system consisting of a polar coordinate system extended by a straight coordinate axis perpendicular to the plane spanned by the polar coordinate system. A CylindricalCS shall have three usesAxis associations.

#### 12.4.4.18 gml:cylindricalCSRef

```
<element name="cylindricalCSRef" type="gml:CylindricalCSRefType"
substitutionGroup="gml:coordinateSystemRef"/>

<complexType name="CylindricalCSRefType">
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:CylindricalCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a cylindrical coordinate system, either referencing or containing the definition of that coordinate system.

#### 12.4.4.19 gml:ObliqueCartesianCS

```
<element name="ObliqueCartesianCS" type="gml:ObliqueCartesianCSType"
substitutionGroup="gml:_CoordinateSystem"/>

<complexType name="ObliqueCartesianCSType">
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
```

```

</complexContent>
</complexType>
```

A two- or three-dimensional coordinate system with straight axes that are not necessarily orthogonal. An ObliqueCartesianCS shall have two or three usesAxis associations.

#### 12.4.4.20 **gml:obliqueCartesianCSRef**

```

<element name="obliqueCartesianCSRef" type="gml:ObliqueCartesianCSRefType"
substitutionGroup="gml:coordinateSystemRef"/>

<complexType name="ObliqueCartesianCSRefType">
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:ObliqueCartesianCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to an oblique-Cartesian coordinate system, either referencing or containing the definition of that coordinate system.

### 12.5 datums schema

#### 12.5.1 Overview

##### 12.5.1.1 Introduction

A datum specifies the relationship of a coordinate system to the earth, or to a moving platform in some Engineering CRSs, thus creating a coordinate reference system. A datum can be used as the basis for one-, two- or three-dimensional systems. Five types of datum are specified: geodetic, vertical, engineering, image, and temporal.

##### 12.5.1.2 Geodetic datum

A geodetic datum is used by a geographic or geocentric CRS, and combines an ellipsoid with a prime meridian. The ellipsoid is used to approximate the shape of the earth, as discussed above in Subclause 12.1.3 and below. A prime meridian defines the origin from which longitude values are specified. Most geodetic datums use Greenwich as their prime meridian.

An ellipsoid is defined to approximate the surface of the geoid. The area for which the approximation is valid varies – traditionally regionally, but with the advent of satellite positioning often globally. One ellipsoid shall be specified with every geodetic datum, even if the ellipsoid is not used computationally. The latter may be the case when a Geocentric CRS is used, e.g., in the calculation of satellite orbit and ground positions from satellite observations. Although use of a Geocentric CRS apparently obviates the need of an ellipsoid, the ellipsoid usually played a role in the determination of the associated geodetic datum.

An ellipsoid is defined either by its semi-major axis and inverse flattening, or by its semi-major axis and semi-minor axis. For some applications, for example small-scale mapping in atlases, a spherical approximation of the geoid's surface is used, requiring only the radius of the sphere to be specified. In the XML Schema, these options are modelled by a mandatory element “semiMajorAxis”, plus a “secondDefiningParameter” element. That element allows specification of the semiMinorAxis or inverseFlattening as the second defining ellipsoid parameter, or can specify that a spherical model is used. For a sphere, the “semiMajorAxis” is interpreted as the radius of the sphere.

### 12.5.1.3 Vertical datum

A vertical datum can only be associated with a vertical coordinate reference system. Further sub-typing is required to describe vertical datums adequately. The following types of vertical datum are distinguished:

- a) **Geoidal.** The zero value of the associated (vertical) coordinate system axis is defined to approximate a constant potential surface, usually the geoid. Such a reference surface is usually determined by a national or scientific authority and is then a well-known, named datum. This is the default vertical datum type, because it is the most common one encountered.
- b) **Depth.** The zero point of the vertical axis is defined by a surface that has meaning for the purpose the associated vertical measurements are used for. For hydrographic charts, this is often a predicted nominal sea surface (i.e., without waves or other wind and current effects) that occurs at low tide. Examples are Lowest Astronomical Tide and Lowest Low Water Spring. A different example is a sloping and undulating River Datum defined as the nominal river water surface occurring at a quantified river discharge.
- c) **Barometric.** A vertical datum is of type “barometric” if atmospheric pressure is the basis for the definition of the origin. Atmospheric pressure may be used as the intermediary to determine height (barometric height determination) or it may be used directly as the vertical ordinate, against which other parameters are measured. The latter case is applied routinely in meteorology.

Barometric height determination is routinely used in aircraft. The altimeter (barometer) on board is set to the altitude of the airfield at the time of take-off, which corrects simultaneously for instantaneous air pressure and altitude of the airfield. The measured height value is commonly named “altitude”.

In some land surveying applications, height differences between points are measured with barometers. To obtain absolute heights, the measured height differences are added to the known heights of control points. In that case the vertical datum type is not barometric, but is the same as that of the vertical control network used to obtain the heights of the new points and its vertical datum type.

The accuracy of this technique is limited, as it is affected strongly by the spatial and temporal variability of atmospheric pressure. This accuracy limitation impacts the precision of the associated vertical datum definition. The datum is usually the surface of constant atmospheric pressure approximately equating to mean sea level (MSL). The origin or anchor point is usually a point of known MSL height. The instruments are calibrated at this point by correcting for the instantaneous atmospheric pressure at sea level and the height of the point above MSL.

In meteorology, atmospheric pressure routinely takes the role of vertical ordinate in a CRS that is used as a spatial reference frame for meteorological parameters in the upper atmosphere. The origin of the datum is the (hypothetical) zero atmospheric pressure and the positive vertical axis points down (to increasing pressure).

- d) **Other surface.** In some cases, e.g. oil exploration and production, geological features, i.e., the top or bottom of a geologically identifiable and meaningful subsurface layer, are sometimes used as a vertical datum. Other variations to the above three vertical datum types may exist and are all bracketed in this category.

### 12.5.1.4 Image datum

An image datum is used in a local context, to describe the origin of an image coordinate reference system. The image grid is defined as the set of lines of constant integer ordinate values. The term “image grid” is often used in other standards to describe the concept of Image CRS. However, care shall be taken to correctly interpret this term in the context in which it is used. The term “grid cell” is often used as a substitute for the term “pixel”.

The grid lines of the image may be associated in two ways with data values of the pixel or grid cell (ISO 19123). The data values of the image usually represent an average or integrated value that is associated with the entire pixel.

- a) An image grid can be associated with data values such that the grid lines run through the centres of the pixels. The cell centres will thus have integer coordinate values. In this case the “pixel in cell” will have the value “cell centre”.
- b) Alternatively, the image grid may be defined such that the grid lines define the cell or pixel corners rather than the cell centres. The cell centres will thus have non-integer coordinate values, the fractional parts always being 0.5. ISO 19123 calls the grid points in this latter case “posts” and associated image data: “matrix data”. The “pixel in cell” will now have the value “cell corner”.

This difference in perspective has no effect on image interpretation, but is important for coordinate transformations involving this image.

#### **12.5.1.5 Engineering datum**

An engineering datum is used in a local context, to describe the origin of an engineering (or local) coordinate reference system.

#### **12.5.1.6 Temporal datum**

A temporal datum is used to describe the origin of a temporal coordinate reference system.

#### **12.5.1.7 Schema introduction**

The datums.xsd schema has three logical parts, which define elements and types for XML encoding of the definitions of:

- a) Abstract datum
- b) Geodetic datums, including ellipsoid and prime meridian
- c) Multiple other concrete types of spatial-temporal datums

This schema encodes the Datum package of the UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex B of Topic 2.

The datums schema includes the referenceSystems.xsd GML schema. This schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd: datums:v3.1.0

#### **12.5.2 Abstract datum**

##### **12.5.2.1 gml:\_Datum**

```

<element name="_Datum" type="gml:AbstractDatumType" abstract="true" substitutionGroup="gml:Definition"/>

<complexType name="AbstractDatumType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractDatumBaseType">
      <sequence>
        <element ref="gml:datumID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:anchorPoint" minOccurs="0"/>
        <element ref="gml:realizationEpoch" minOccurs="0"/>
        <element ref="gml:validArea" minOccurs="0"/>
        <element ref="gml:scope" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

</complexType>

<complexType name="AbstractDatum BaseType" abstract="true">
  <complexContent>
    <restriction base="gml:DefinitionType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:datumName"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>

```

A datum specifies the relationship of a coordinate system to the earth, thus creating a coordinate reference system. A datum uses a parameter or set of parameters that determine the location of the origin of the coordinate reference system. Each datum subtype can be associated with only specific types of coordinate systems. This abstract complexType shall not be used, extended, or restricted, in an Application Schema, to define a concrete subtype with a meaning equivalent to a concrete subtype specified in this document.

The AbstractDatum BaseType provides the basic encoding for datum objects, simplifying and restricting the DefinitionType as needed.

The "datumID" elements contain a set of alternative identifications of this datum. The first datumID, if any, is normally the primary identification code, and any others are aliases.

The "remarks" element contains comments on or information about this reference system, including source information.

#### 12.5.2.2 gml:datumName

```
<element name="datumName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which this datum is identified.

#### 12.5.2.3 gml:datumID

```
<element name="datumID" type="gml:IdentifierType"/>
```

An identification of a datum.

#### 12.5.2.4 gml:anchorPoint

```
<element name="anchorPoint" type="gml:CodeType"/>
```

Description, possibly including coordinates, of the point or points used to anchor the datum to the Earth. Also known as the "origin", especially for engineering and image datums. The codeSpace attribute can be used to reference a source of more detailed on this point or surface, or on a set of such descriptions.

- a) For a geodetic datum, this point is also known as the fundamental point, which is traditionally the point where the relationship between geoid and ellipsoid is defined. In some cases, the "fundamental point" may consist of a number of points. In those cases, the parameters defining the geoid/ellipsoid relationship have been averaged for these points, and the averages adopted as the datum definition.
- b) For an engineering datum, the anchor point may be a physical point, or it may be a point with defined coordinates in another CRS. When appropriate, the coordinates of this anchor point can be referenced in another document, such as referencing a GML feature that references or includes a point position.
- c) For an image datum, the anchor point is usually either the centre of the image or the corner of the image.

- d) For a temporal datum, this attribute is not defined. Instead of the anchor point, a temporal datum carries a separate time origin of type `DateTime`.

#### **12.5.2.5 `gml:realizationEpoch`**

```
<element name="realizationEpoch" type="date"/>
```

The time after which this datum definition is valid. This time may be precise (e.g. 1997.0 for IRTF97) or merely a year (e.g. 1983 for NAD83). In the latter case, the epoch usually refers to the year in which a major recalculation of the geodetic control network, underlying the datum, was executed or initiated. An old datum can remain valid after a new datum is defined. Alternatively, a datum may be superseded by a later datum, in which case the realization epoch for the new datum defines the upper limit for the validity of the superseded datum.

#### **12.5.2.6 `gml:datumRef`**

```
<element name="datumRef" type="gml:DatumRefType" substitutionGroup="gml:dictionaryEntry"/>

<complexType name="DatumRefType">
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:_Datum" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a datum, either referencing or containing the definition of that datum.

### **12.5.3 Geodetic datum**

#### **12.5.3.1 `gml:GeodeticDatum`**

```
<element name="GeodeticDatum" type="gml:GeodeticDatumType" substitutionGroup="gml:_Datum"/>

<complexType name="GeodeticDatumType">
  <complexContent>
    <extension base="gml:AbstractDatumType">
      <sequence>
        <element ref="gml:usesPrimeMeridian"/>
        <element ref="gml:usesEllipsoid"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A geodetic datum defines the precise location and orientation in 3-dimensional space of a defined ellipsoid (or sphere), or of a Cartesian coordinate system centered in this ellipsoid (or sphere).

#### **12.5.3.2 `gml:usesPrimeMeridian`**

```
<element name="usesPrimeMeridian" type="gml:PrimeMeridianRefType"/>
```

Association to the prime meridian used by this geodetic datum.

#### **12.5.3.3 `gml:usesEllipsoid`**

```
<element name="usesEllipsoid" type="gml:EllipsoidRefType"/>
```

Association to the ellipsoid used by this geodetic datum.

#### 12.5.3.4 **gml:geodeticDatumRef**

```
<element name="geodeticDatumRef" type="gml:GeodeticDatumRefType" substitutionGroup="gml:datumRef"/>

<complexType name="GeodeticDatumRefType">
  <complexContent>
    <restriction base="gml:DatumRefType">
      <sequence>
        <element ref="gml:GeodeticDatum" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a geodetic datum, either referencing or containing the definition of that datum.

#### 12.5.3.5 **gml:Ellipsoid**

```
<element name="Ellipsoid" type="gml:EllipsoidType" substitutionGroup="gml:Definition"/>

<complexType name="EllipsoidType">
  <complexContent>
    <extension base="gml:Ellipsoid BaseType">
      <sequence>
        <element ref="gml:ellipsoidID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:semiMajorAxis"/>
        <element ref="gml:secondDefiningParameter"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="Ellipsoid BaseType" abstract="true">
  <complexContent>
    <restriction base="gml:DefinitionType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:ellipsoidName"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
```

An ellipsoid is a geometric figure that can be used to describe the approximate shape of the earth. In mathematical terms, it is a surface formed by the rotation of an ellipse about its minor axis.

The EllipsoidBaseType provides the basic encoding for ellipsoid objects, simplifying and restricting the DefinitionType as needed.

The "ellipsoidID" elements contain a set of alternative identifications of this ellipsoid. The first ellipsoidID, if any, is normally the primary identification code, and any others are aliases.

The "remarks" element contains comments on or information about this reference system, including source information.

#### **12.5.3.6 `gml:ellipsoidName`**

```
<element name="ellipsoidName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which this ellipsoid is identified.

#### **12.5.3.7 `gml:ellipsoidID`**

```
<element name="ellipsoidID" type="gml:IdentifierType"/>
```

An identification of an ellipsoid.

#### **12.5.3.8 `gml:semiMajorAxis`**

```
<element name="semiMajorAxis" type="gml:LengthType"/>
```

Length of the semi-major axis of the ellipsoid.

#### **12.5.3.9 `gml:ellipsoidRef`**

```
<element name="ellipsoidRef" type="gml:EllipsoidRefType" substitutionGroup="gml:dictionaryEntry"/>

<complexType name="EllipsoidRefType">
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:Ellipsoid" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to an ellipsoid, either referencing or containing the definition of that ellipsoid.

#### **12.5.3.10 `gml:secondDefiningParameter`**

```
<element name="secondDefiningParameter" type="gml:SecondDefiningParameterType"/>

<complexType name="SecondDefiningParameterType">
  <choice>
    <element ref="gml:inverseFlattening"/>
    <element ref="gml:semiMinorAxis"/>
    <element ref="gml:isSphere"/>
  </choice>
</complexType>
```

Definition of the second parameter that defines the shape of an ellipsoid. An ellipsoid requires two defining parameters: semi-major axis and inverse flattening or semi-major axis and semi-minor axis. When the reference body is a sphere rather than an ellipsoid, only a single defining parameter is required, namely the radius of the sphere; in that case, the semi-major axis "degenerates" into the radius of the sphere.

#### **12.5.3.11 `gml:inverseFlattening`**

```
<element name="inverseFlattening" type="gml:ScaleType"/>
```

Inverse flattening value of the ellipsoid

### 12.5.3.12 **gml:semiMinorAxis**

```
<element name="semiMinorAxis" type="gml:LengthType"/>
```

Length of the semi-minor axis of the ellipsoid.

### 12.5.3.13 **gml:isSphere**

```
<element name="isSphere">
  <simpleType>
    <restriction base="string">
      <enumeration value="sphere"/>
    </restriction>
  </simpleType>
</element>
```

When this element is included, the ellipsoid is degenerate and is actually a sphere. The sphere is completely defined by the semi-major axis, which is the radius of the sphere.

### 12.5.3.14 **gml:PrimeMeridian**

```
<element name="PrimeMeridian" type="gml:PrimeMeridianType" substitutionGroup="gml:Definition"/>

<complexType name="PrimeMeridianType">
  <complexContent>
    <extension base="gml:PrimeMeridian BaseType">
      <sequence>
        <element ref="gml:meridianID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:greenwichLongitude"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="PrimeMeridian BaseType" abstract="true">
  <complexContent>
    <restriction base="gml:DefinitionType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:meridianName"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
```

A prime meridian defines the origin from which longitude values are determined.

The PrimeMeridianBaseType provides the basic encoding for prime meridian objects, simplifying and restricting the DefinitionType as needed.

The "meridianID" elements contain a set of alternative identifications of this prime meridian. The first meridianID, if any, is normally the primary identification code, and any others are aliases.

The "remarks" element contains comments on or information about this reference system, including source information.

### 12.5.3.15 **gml:meridianName**

```
<element name="meridianName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which this prime meridian is identified. The meridianName most common value is “Greenwich”, and that value shall be used when the greenwichLongitude value is zero.

#### 12.5.3.16 **gml:meridianID**

```
<element name="meridianID" type="gml:IdentifierType"/>
```

An identification of a prime meridian.

#### 12.5.3.17 **gml:greenwichLongitude**

```
<element name="greenwichLongitude" type="gml:AngleChoiceType"/>
```

Longitude of the prime meridian measured from the Greenwich meridian, positive eastward. The greenwichLongitude most common value is zero, and that value shall be used when the meridianName value is “Greenwich”.

#### 12.5.3.18 **gml:primeMeridianRef**

```
<element name="primeMeridianRef" type="gml:PrimeMeridianRefType" substitutionGroup="gml:dictionaryEntry"/>

<complexType name="PrimeMeridianRefType">
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:PrimeMeridian" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a prime meridian, either referencing or containing the definition of that meridian.

### 12.5.4 Other concrete datums

#### 12.5.4.1 **gml:EngineeringDatum**

```
<element name="EngineeringDatum" type="gml:EngineeringDatumType" substitutionGroup="gml:_Datum"/>

<complexType name="EngineeringDatumType">
  <complexContent>
    <extension base="gml:AbstractDatumType"/>
  </complexContent>
</complexType>
```

An engineering datum defines the origin of an engineering coordinate reference system, and is used in a region around that origin. This origin can be fixed with respect to the earth (such as a defined point at a construction site), or be a defined point on a moving vehicle (such as on a ship or satellite).

#### 12.5.4.2 **gml:engineeringDatumRef**

```
<element name="engineeringDatumRef" type="gml:EngineeringDatumRefType" substitutionGroup="gml:datumRef"/>

<complexType name="EngineeringDatumRefType">
  <complexContent>
    <restriction base="gml:DatumRefType">
      <sequence>
        <element ref="gml:EngineeringDatum" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
```

```

        <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
</complexContent>
</complexType>
```

Association to an engineering datum, either referencing or containing the definition of that datum.

#### 12.5.4.3 gml:ImageDatum

```

<element name="ImageDatum" type="gml:ImageDatumType" substitutionGroup="gml:_Datum"/>

<complexType name="ImageDatumType">
    <complexContent>
        <extension base="gml:AbstractDatumType">
            <sequence>
                <element ref="gml:pixelInCell"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

An image datum defines the origin of an image coordinate reference system, and is used in a local context only. For more information, see Subclause 12.5.1.

#### 12.5.4.4 gml:pixelInCell

```

<element name="pixelInCell" type="gml:PixelInCellType"/>

<complexType name="PixelInCellType">
    <simpleContent>
        <restriction base="gml:CodeType">
            <attribute name="codeSpace" type="anyURI" use="required"/>
        </restriction>
    </simpleContent>
</complexType>
```

Specification of the way an image grid is associated with the image data attributes. The required codeSpace attribute shall reference a source of information specifying the values and meanings of all the allowed string values for this PixelInCellType.

#### 12.5.4.5 gml:imageDatumRef

```

<element name="imageDatumRef" type="gml:ImageDatumRefType" substitutionGroup="gml:datumRef"/>

<complexType name="ImageDatumRefType">
    <complexContent>
        <restriction base="gml:DatumRefType">
            <sequence>
                <element ref="gml:ImageDatum" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
```

Association to an image datum, either referencing or containing the definition of that datum.

#### 12.5.4.6 gml:VerticalDatum

```

<element name="VerticalDatum" type="gml:VerticalDatumType" substitutionGroup="gml:_Datum"/>

<complexType name="VerticalDatumType">
    <complexContent>
```

```

<extension base="gml:AbstractDatumType">
  <sequence>
    <element ref="gml:verticalDatumType" minOccurs="0"/>
  </sequence>
</extension>
</complexContent>
</complexType>

```

A textual description and/or a set of parameters identifying a particular reference level surface used as a zero-height surface, including its position with respect to the Earth for any of the height types recognized by this standard. There are several types of Vertical Datums, and each may place constraints on the Coordinate Axis with which it is combined to create a Vertical CRS.

#### 12.5.4.7 gml:verticalDatumType

```

<element name="verticalDatumType" type="gml:VerticalDatumTypeType"/>

<complexType name="VerticalDatumTypeType">
  <simpleContent>
    <restriction base="gml:CodeType">
      <attribute name="codeSpace" type="anyURI" use="required"/>
    </restriction>
  </simpleContent>
</complexType>

```

Type of a vertical datum. The required codeSpace attribute shall reference a source of information specifying the values and meanings of all the allowed string values for this VerticalDatumTypeType.

#### 12.5.4.8 gml:verticalDatumRef

```

<element name="verticalDatumRef" type="gml:VerticalDatumRefType" substitutionGroup="gml:datumRef"/>

<complexType name="VerticalDatumRefType">
  <complexContent>
    <restriction base="gml:DatumRefType">
      <sequence>
        <element ref="gml:VerticalDatum" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a vertical datum, either referencing or containing the definition of that datum.

#### 12.5.4.9 gml:TemporalDatum

```

<element name="TemporalDatum" type="gml:TemporalDatumType" substitutionGroup="gml:_Datum"/>

<complexType name="TemporalDatumType">
  <complexContent>
    <extension base="gml:TemporalDatumBaseType">
      <sequence>
        <element ref="gml:origin"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="TemporalDatumBaseType" abstract="true">
  <complexContent>
    <restriction base="gml:AbstractDatumType">
      <sequence>

```

```

<element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:datumName"/>
<element ref="gml:datumID" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:validArea" minOccurs="0"/>
<element ref="gml:scope" minOccurs="0"/>
</sequence>
<attribute ref="gml:id" use="required"/>
</restriction>
</complexContent>
</complexType>

```

A temporal datum defines the origin of a temporal coordinate reference system. This type extends the TemporalDatumRestrictionType to add the "origin" element with the dateTime type.

The TemporalDatumBaseType partially defines the origin of a temporal coordinate reference system. This type restricts the AbstractDatumType to remove the "anchorPoint" and "realizationEpoch" elements.

#### 12.5.4.10 gml:origin

```
<element name="origin" type="dateTime"/>
```

The date and time origin of this temporal datum.

#### 12.5.4.11 gml:temporalDatumRef

```

<element name="temporalDatumRef" type="gml:TemporalDatumRefType" substitutionGroup="gml:datumRef"/>

<complexType name="TemporalDatumRefType">
<complexContent>
<restriction base="gml:DatumRefType">
<sequence>
<element ref="gml:TemporalDatum" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</restriction>
</complexContent>
</complexType>

```

Association to a temporal datum, either referencing or containing the definition of that datum.

### 12.6 coordinateOperations schema

#### 12.6.1 Overview

##### 12.6.1.1 Introduction

A coordinate operation defines the coordinate transformation or conversion between coordinates in two different coordinate reference systems. Provided the relationship between any two coordinate reference systems is known, coordinates can be transformed or converted to another coordinate reference system.

A coordinate operation therefore usually specifies a source and target coordinate reference system. For that reason a coordinate operation is often popularly said to operate between two coordinate reference systems. Although this wording may be good enough for conversation, it must be realised that coordinate operations do not operate on coordinate reference systems, but on coordinates. This fact is important because of its implication that coordinate reference systems cannot be 'created' from another coordinate reference system by means of a coordinate operation. Neither can a coordinate operation be used to modify the definition of a coordinate reference system, e.g., by converting the units of measure of the coordinates.

### **12.6.1.2 Coordinate operation types**

Coordinate operations are divided into four types:

- a) Coordinate conversion – mathematical operation on coordinates that does not include any change of Datum. The best-known example of a coordinate conversion is a map projection. The parameters describing coordinate conversions are defined rather than empirically derived. Please note that some conversions have no parameters.
- b) Coordinate transformation – mathematical operation on coordinates that includes a change of Datum. The parameters of a coordinate transformation are empirically derived from a dataset containing the coordinates of a series of points in both coordinate reference systems. This computational process is usually ‘overdetermined’, allowing derivation of error (or accuracy) estimates for the transformation. Also, the stochastic nature of the parameters may result in multiple (different) instantiations of the same coordinate transformation.
- c) Concatenated operation – sequential application of two or more coordinate transformations and/or conversions.
- d) Pass through operation – application of a specified coordinate conversion or transformation to convert a subset of the coordinates of a compound CRS.

A coordinate transformation or conversion can be time-varying, and shall be time-varying if either the source or target CRS is time varying. When the coordinate operation is time-varying, the operation method used will also be time-varying, and some of the parameters used by that operation method will involve time. For example, some of the parameters may have time, speed, and/or acceleration values and units. The variables used in the operation method algorithm may include time explicitly or only implicitly (for example, time may be an unspecified function of other variables).

### **12.6.1.3 Coordinate conversions**

Coordinate conversions are coordinate operations that make use of exact, defined (rather than measured or computed), and therefore error-free parameter values. Corresponding pairs of coordinate tuples in each of the two coordinate reference systems connected through a coordinate conversion have a fixed arithmetic relationship. Additionally one of the two tuples cannot exist without specification of the coordinate conversion and the ‘source’ coordinate reference system. Coordinate conversions are therefore intimately related to the earlier-defined concept of Derived CRSs.

The best-known example of this source-derived relationship is a projected coordinate reference system, which is always based on a source geographic coordinate reference system. The associated map projection effectively defines the projected coordinate reference system from the geographic coordinate system. This concept is modelled as a direct link between coordinate reference system and coordinate conversion.

Please note that this does not contradict the statement above that a coordinate operation cannot be used to create or modify a coordinate reference system in a software implementation. The text above describes a static source-result relationship between coordinates in the two coordinate reference systems. For such defining coordinate conversions, no source and target coordinate reference system are defined. The usage of such a coordinate conversion is in the coordinate reference system, which will point to that conversion and to its source coordinate reference system.

The source and target coordinate reference system of a coordinate conversion are defined in the GeneralDerivedCRS, by specifying the base (i.e., source) CRS and the defining conversion. The derived coordinate reference system itself is the target CRS in this situation.

### **12.6.1.4 Concatenated coordinate operation**

A concatenated coordinate operation is an ordered sequence of coordinate operations. The sequence of operations is constrained by the requirement that the source coordinate reference system of step  $(n+1)$  shall

be the same as the target coordinate reference system of step ( $n$ ). The source coordinate reference system of the first step and the target coordinate reference system of the last step are the source and target coordinate reference system associated with the concatenated operation.

The above constraint should not be interpreted as implying that only those coordinate operations can be used in a concatenated operation that have their source and a target coordinate reference system specified through the association pair between a CoordinateOperation and a CRS. This would exclude coordinate conversions. Concatenated coordinate operations may contain coordinate transformations and/or coordinate conversions. When used in a concatenated operation, the conversion's source and target coordinate reference system are equally subject to the above constraint as the source and target of a transformation although they are specified in a different manner.

#### 12.6.1.5 Pass-through coordinate operation

Coordinate operations require input coordinate tuples of certain dimensions, and produce output tuples of certain dimensions. The dimensions of these coordinate tuples and the dimensions of the coordinate reference system they are defined in shall be the same.

The ability to define compound coordinate reference systems combining two or more other coordinate reference systems, not themselves compound, introduces a difficulty. For example, it may be required to transform only the horizontal or only the vertical component of a compound coordinate reference system, which will put them at odds with coordinate operations specified for either horizontal or vertical coordinates only. To the human mind this is a trivial problem, but not so for coordinate transformation software that ought to be capable of automatic operation, without human intervention. The software logic would be confronted with the problem of having to apply a 2-dimensional coordinate operation to 3-dimensional coordinate tuples.

This problem has been solved by defining a pass-through operation. This operation specifies what subset of a coordinate tuple is subject to a requested transformation. It takes the form of referencing another coordinate operation and specifying a sequence of numbers defining the positions in the coordinate tuple of the coordinates affected by that transformation.

NOTE. The order of the coordinates in a coordinate tuple shall agree with the order of the coordinate system axes as defined for the associated coordinate system.

#### 12.6.1.6 Operation method and parameters

The algorithm used to execute the coordinate operation is defined in the operation method. Concatenated operations and pass-through operations do not require a coordinate operation to be specified. Each operation method uses a number of parameters (although some coordinate conversions use none), and each coordinate operation assigns value to these parameters.

Most parameter values are numeric, but for some operation methods, notably those implementing a grid interpolation algorithm, the parameter value could be a file name and location (this may be a URI). An example is the coordinate transformation from NAD 27 to NAD 83 in the USA; depending on the locations of the points to be transformed, one of a series of grid files should be used.

As this class comes close to the heart of any coordinate transformation software, it is recommended to make extensive use of identifiers, referencing well-known datasets wherever possible. There is as yet no standard way of spelling or even naming the various coordinate operation methods. Client software requesting a coordinate operation to be executed by a coordinate transformation server implementation may therefore ask for an operation method this server doesn't recognise, although a perfectly valid method may be available. The same holds for coordinate operation parameters used by any coordinate operation method.

To facilitate recognition and validation, the operation formulae shall be included or referenced in an operation method, and if possible a worked example.

Some operation methods may require a large number of operation parameters. Also, some operation methods require that groups of parameters be repeatable as a group. In such cases, it is helpful to group related parameters in parameter groups. Two or more parameter groups are then associated with a particular

coordinate operation method, and each parameter group consists of a set of operation parameters, or of other nested parameter groups. This way of modelling is not mandatory; all operation parameters can be directly associated with the operation method.

#### 12.6.1.7 Schema introduction

The spatial-temporal coordinate operations schema, coordinateOperations.xsd, has five logical parts, which define elements and types for XML encoding of the definitions of:

- a) Multiple abstract coordinate operations
- b) Multiple concrete types of coordinate operations, including Transformations and Conversions
- c) Abstract and concrete parameter values and groups
- d) Operation methods
- e) Abstract and concrete operation parameters and groups

This schema encodes the Coordinate Operation package of the UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex B of Topic 2.

The coordinateOperations schema includes the coordinateReferenceSystems.xsd and dataQuality.xsd GML schemas. This schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd: coordinateOperations:v3.1.0

#### 12.6.2 Abstract coordinate operations

##### 12.6.2.1 gml:\_CoordinateOperation

```
<element name="_CoordinateOperation" type="gml:AbstractCoordinateOperationType" abstract="true"
substitutionGroup="gml:Definition"/>

<complexType name="AbstractCoordinateOperationType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractCoordinateOperationBaseType">
      <sequence>
        <element ref="gml:coordinateOperationID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:operationVersion" minOccurs="0"/>
        <element ref="gml:validArea" minOccurs="0"/>
        <element ref="gml:scope" minOccurs="0"/>
        <element ref="gml:_positionalAccuracy" minOccurs="0"/>
        <element ref="gml:sourceCRS" minOccurs="0"/>
        <element ref="gml:targetCRS" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="AbstractCoordinateOperationBaseType" abstract="true">
  <complexContent>
    <restriction base="gml:DefinitionType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:coordinateOperationName"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
```

```
</complexContent>
</complexType>
```

A mathematical operation on coordinates that transforms or converts coordinates to another coordinate reference system. Many but not all coordinate operations (from CRS A to CRS B) also uniquely define the inverse operation (from CRS B to CRS A). In some cases, the operation method algorithm for the inverse operation is the same as for the forward algorithm, but the signs of some operation parameter values shall be reversed. In other cases, different algorithms are required for the forward and inverse operations, but the same operation parameter values are used. If (some) entirely different parameter values are needed, a different coordinate operation shall be defined.

The AbstractCoordinateOperationBaseType provides the basic encoding for coordinate operation objects, simplifying and restricting the DefinitionType as needed.

The optional \_positionalAccuracy elements provide an unordered set of estimates of the impact of this coordinate operation on point position accuracy. This element provides position error estimates for target coordinates of this coordinate operation, assuming no errors in source coordinates.

The "coordinateOperationID" elements contain a set of alternative identifications of this coordinate operation. The first coordinateOperationID, if any, is normally the primary identification code, and any others are aliases.

The "remarks" element contains comments on or information about this reference system, including source information.

#### **12.6.2.2   gml:coordinateOperationName**

```
<element name="coordinateOperationName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which this coordinate operation is identified.

#### **12.6.2.3   gml:coordinateOperationID**

```
<element name="coordinateOperationID" type="gml:IdentifierType"/>
```

An identification of a coordinate operation.

#### **12.6.2.4   gml:operationVersion**

```
<element name="operationVersion" type="string"/>
```

Version of the coordinate transformation (i.e., instantiation due to the stochastic nature of the parameters). Mandatory when describing a transformation, and should not be supplied for a conversion.

#### **12.6.2.5   gml:sourceCRS**

```
<element name="sourceCRS" type="gml:CRSRefType"/>
```

Association to the source CRS (coordinate reference system) of this coordinate operation.

#### **12.6.2.6   gml:targetCRS**

```
<element name="targetCRS" type="gml:CRSRefType"/>
```

Association to the target CRS (coordinate reference system) of this coordinate operation. For constraints on multiplicity of "sourceCRS" and "targetCRS", see Subclause 12.6.1.

#### **12.6.2.7   gml:coordinateOperationRef**

```
<element name="coordinateOperationRef" type="gml:CoordinateOperationRefType"
```

```

substitutionGroup="gml:dictionaryEntry"/>

<complexType name="CoordinateOperationRefType">
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:_CoordinateOperation" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a coordinate operation, either referencing or containing the definition of that coordinate operation.

#### **12.6.2.8 gml:\_SingleOperation**

```

<element name="_SingleOperation" type="gml:AbstractSingleOperationType" abstract="true"
substitutionGroup="gml:_CoordinateOperation"/>

<complexType name="AbstractSingleOperationType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractCoordinateOperationType"/>
  </complexContent>
</complexType>

```

A single (not concatenated) coordinate operation.

#### **12.6.2.9 gml:singleOperationRef**

```

<element name="singleOperationRef" type="gml:SingleOperationRefType"
substitutionGroup="gml:coordinateOperationRef"/>

<complexType name="SingleOperationRefType">
  <complexContent>
    <restriction base="gml:CoordinateOperationRefType">
      <sequence>
        <element ref="gml:_SingleOperation" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a single operation, either referencing or containing the definition of that single operation.

#### **12.6.2.10 gml:\_Operation**

```

<element name="_Operation" type="gml:AbstractOperationType" abstract="true"
substitutionGroup="gml:_SingleOperation"/>

<complexType name="AbstractOperationType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractSingleOperationType"/>
  </complexContent>
</complexType>

```

A parameterized mathematical operation on coordinates that transforms or converts coordinates to another coordinate reference system. This coordinate operation uses an operation method, usually with associated parameter values. However, operation methods and parameter values are directly associated with concrete subtypes, not with this abstract type.

This abstract complexType shall not be directly used, extended, or restricted in a compliant Application Schema.

#### 12.6.2.11 **gml:operationRef**

```
<element name="operationRef" type="gml:OperationRefType" substitutionGroup="gml:singleOperationRef"/>

<complexType name="OperationRefType">
  <complexContent>
    <restriction base="gml:SingleOperationRefType">
      <sequence>
        <element ref="gml:_Operation" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to an abstract operation, either referencing or containing the definition of that operation.

#### 12.6.2.12 **gml:\_GeneralConversion**

```
<element name="_GeneralConversion" type="gml:AbstractGeneralConversionType" abstract="true"
substitutionGroup="gml:_Operation"/>

<complexType name="AbstractGeneralConversionType" abstract="true">
  <complexContent>
    <restriction base="gml:AbstractOperationType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:coordinateOperationName"/>
        <element ref="gml:coordinateOperationID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:validArea" minOccurs="0"/>
        <element ref="gml:scope" minOccurs="0"/>
        <element ref="gml:_positionalAccuracy" minOccurs="0"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
```

An abstract operation on coordinates that does not include any change of datum. The best-known example of a coordinate conversion is a map projection. The parameters describing coordinate conversions are defined rather than empirically derived. Note that some conversions have no parameters.

This abstract complexType is expected to be extended for well-known operation methods with many Conversion instances, in Application Schemas that define operation-method-specialized element names and contents. This conversion uses an operation method, usually with associated parameter values. However, operation methods and parameter values are directly associated with concrete subtypes, not with this abstract type. All concrete types derived from this type shall extend this type to include a "usesMethod" element that references the "OperationMethod" element. Similarly, all concrete types derived from this type shall extend this type to include zero or more elements each named "uses...Value" that each use the type of an element substitutable for the "\_generalParameterValue" element.

#### 12.6.2.13 **gml:generalConversionRef**

```
<element name="generalConversionRef" type="gml:GeneralConversionRefType"
substitutionGroup="gml:operationRef"/>

<complexType name="GeneralConversionRefType">
  <complexContent>
    <restriction base="gml:OperationRefType">
```

```

<sequence>
    <element ref="gml:_GeneralConversion" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</restriction>
</complexContent>
</complexType>

```

Association to a general conversion, either referencing or containing the definition of that conversion.

#### 12.6.2.14 gml:\_GeneralTransformation

```

<element name="_GeneralTransformation" type="gml:AbstractGeneralTransformationType" abstract="true"
substitutionGroup="gml:_Operation"/>

<complexType name="AbstractGeneralTransformationType" abstract="true">
    <complexContent>
        <restriction base="gml:AbstractOperationType">
            <sequence>
                <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:remarks" minOccurs="0"/>
                <element ref="gml:coordinateOperationName"/>
                <element ref="gml:coordinateOperationID" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:operationVersion"/>
                <element ref="gml:validArea" minOccurs="0"/>
                <element ref="gml:scope" minOccurs="0"/>
                <element ref="gml:_positionalAccuracy" minOccurs="0"/>
                <element ref="gml:sourceCRS"/>
                <element ref="gml:targetCRS"/>
            </sequence>
            <attribute ref="gml:id" use="required"/>
        </restriction>
    </complexContent>
</complexType>

```

An abstract operation on coordinates that usually includes a change of Datum. The parameters of a coordinate transformation are empirically derived from data containing the coordinates of a series of points in both coordinate reference systems. This computational process is usually "over-determined", allowing derivation of error (or accuracy) estimates for the transformation. Also, the stochastic nature of the parameters may result in multiple (different) versions of the same coordinate transformation.

This abstract complexType is expected to be extended for well-known operation methods with many Transformation instances, in Application Schemas that define operation-method-specialized value element names and contents. This transformation uses an operation method with associated parameter values. However, operation methods and parameter values are directly associated with concrete subtypes, not with this abstract type. All concrete types derived from this type shall extend this type to include a "usesMethod" element that references one "OperationMethod" element. Similarly, all concrete types derived from this type shall extend this type to include one or more elements each named "uses...Value" that each use the type of an element substitutable for the "\_generalParameterValue" element.

#### 12.6.2.15 gml:generalTransformationRef

```

<element name="generalTransformationRef" type="gml:GeneralTransformationRefType"
substitutionGroup="gml:operationRef"/>

<complexType name="GeneralTransformationRefType">
    <complexContent>
        <restriction base="gml:OperationRefType">
            <sequence>
                <element ref="gml:_GeneralTransformation" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>

```

```

        </restriction>
    </complexContent>
</complexType>
```

Association to a general transformation, either referencing or containing the definition of that transformation.

### 12.6.3 Concrete coordinate operations

#### 12.6.3.1 gml:ConcatenatedOperation

```

<element name="ConcatenatedOperation" type="gml:ConcatenatedOperationType"
substitutionGroup="gml:_CoordinateOperation"/>

<complexType name="ConcatenatedOperationType">
    <complexContent>
        <extension base="gml:AbstractCoordinateOperationType">
            <sequence>
                <element ref="gml:usesSingleOperation" minOccurs="2" maxOccurs="unbounded"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

A concatenated operation is an ordered sequence of two or more single coordinate operations. The sequence of operations is constrained by the requirement that the source coordinate reference system of step (n+1) shall be the same as the target coordinate reference system of step (n). The source coordinate reference system of the first step and the target coordinate reference system of the last step are the source and target coordinate reference system associated with the concatenated operation. Instead of a forward operation, an inverse operation may be used for one or more of the operation steps mentioned above, if the inverse operation is uniquely defined by the forward operation.

The usesSingleOperation elements are an ordered sequence of associations to the two or more single operations used by this concatenated operation.

#### 12.6.3.2 gml:usesSingleOperation

```
<element name="usesSingleOperation" type="gml:SingleOperationRefType"/>
```

Association to a single operation.

#### 12.6.3.3 gml:concatenatedOperationRef

```

<element name="concatenatedOperationRef" type="gml:ConcatenatedOperationRefType"
substitutionGroup="gml:coordinateOperationRef"/>

<complexType name="ConcatenatedOperationRefType">
    <complexContent>
        <restriction base="gml:CoordinateOperationRefType">
            <sequence>
                <element ref="gml:ConcatenatedOperation" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
```

Association to a concatenated operation, either referencing or containing the definition of that concatenated operation.

#### 12.6.3.4 gml:PassThroughOperation

```
<element name="PassThroughOperation" type="gml:PassThroughOperationType"
```

```

substitutionGroup="gml:_SingleOperation"/>

<complexType name="PassThroughOperationType">
  <complexContent>
    <extension base="gml:AbstractSingleOperationType">
      <sequence>
        <element ref="gml:modifiedCoordinate" maxOccurs="unbounded"/>
        <element ref="gml:usesOperation"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

A pass-through operation specifies that a subset of a coordinate tuple is subject to a specific coordinate operation.

The modifiedCoordinate elements are an ordered sequence of positive integers defining the positions in a coordinate tuple of the coordinates affected by this pass-through operation.

#### 12.6.3.5 gml:modifiedCoordinate

```
<element name="modifiedCoordinate" type="positiveInteger"/>
```

#### 12.6.3.6 A positive integer defining a position in a coordinate tuple.gml:usesOperation

```
<element name="usesOperation" type="gml:OperationRefType"/>
```

Association to the operation applied to the specified ordinates.

#### 12.6.3.7 gml:passThroughOperationRef

```

<element name="passThroughOperationRef" type="gml:PassThroughOperationRefType"
substitutionGroup="gml:singleOperationRef"/>

<complexType name="PassThroughOperationRefType">
  <complexContent>
    <restriction base="gml:SingleOperationRefType">
      <sequence>
        <element ref="gml:PassThroughOperation" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

Association to a pass through operation, either referencing or containing the definition of that pass through operation.

#### 12.6.3.8 gml:Conversion

```

<element name="Conversion" type="gml:ConversionType" substitutionGroup="gml:_GeneralConversion"/>

<complexType name="ConversionType">
  <complexContent>
    <extension base="gml:AbstractGeneralConversionType">
      <sequence>
        <element ref="gml:usesMethod"/>
        <element ref="gml:usesValue" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

A concrete operation on coordinates that does not include any change of Datum. The best-known example of a coordinate conversion is a map projection. The parameters describing coordinate conversions are defined rather than empirically derived. Note that some conversions have no parameters.

This concrete complexType can be used with all operation methods, without using an Application Schema that defines operation-method-specialized element names and contents, especially for methods with only one Conversion instance.

The usesValue elements are an unordered list of composition associations to the set of parameter values used by this conversion operation.

#### 12.6.3.9 **gml:usesMethod**

```
<element name="usesMethod" type="gml:OperationMethodRefType"/>
```

Association to the operation method used by this coordinate operation.

#### 12.6.3.10 **gml:usesValue**

```
<element name="usesValue" type="gml:ParameterValueType"/>
```

#### 12.6.3.11 **Composition association to a parameter value used by this coordinate operation.gml:conversionRef**

```
<element name="conversionRef" type="gml:ConversionRefType" substitutionGroup="gml:generalConversionRef"/>

<complexType name="ConversionRefType">
  <complexContent>
    <restriction base="gml:GeneralConversionRefType">
      <sequence>
        <element ref="gml:Conversion" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a concrete general-purpose conversion, either referencing or containing the definition of that conversion.

#### 12.6.3.12 **gml:Transformation**

```
<element name="Transformation" type="gml:TransformationType"
substitutionGroup="gml:_GeneralTransformation"/>

<complexType name="TransformationType">
  <complexContent>
    <extension base="gml:AbstractGeneralTransformationType">
      <sequence>
        <element ref="gml:usesMethod"/>
        <element ref="gml:usesValue" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A concrete operation on coordinates that usually includes a change of datum. The parameters of a coordinate transformation are empirically derived from data containing the coordinates of a series of points in both coordinate reference systems. This computational process is usually "over-determined", allowing derivation of error (or accuracy) estimates for the transformation. Also, the stochastic nature of the parameters may result in multiple (different) versions of the same coordinate transformation.

This concrete complexType can be used for operation methods, without using an Application Schema that defines operation-method-specialized element names and contents, especially for methods with only one Transformation instance.

The usesValue elements are an unordered list of composition associations to the set of parameter values used by this transformation operation.

#### 12.6.3.13 **gml:transformationRef**

```
<element name="transformationRef" type="gml:TransformationRefType"
substitutionGroup="gml:generalTransformationRef"/>

<complexType name="TransformationRefType">
  <complexContent>
    <restriction base="gml:GeneralTransformationRefType">
      <sequence>
        <element ref="gml:Transformation" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a transformation, either referencing or containing the definition of that transformation.

### 12.6.4 Parameter values and groups

#### 12.6.4.1 **gml:\_generalParameterValue**

```
<element name="_generalParameterValue" type="gml:AbstractGeneralParameterValueType" abstract="true"/>

<complexType name="AbstractGeneralParameterValueType" abstract="true">
  <sequence/>
</complexType>
```

Abstract parameter value or group of parameter values.

This abstract complexType is expected to be extended and restricted for well-known operation methods with many instances, in Application Schemas that define operation-method-specialized element names and contents. Specific parameter value elements are directly contained in concrete subtypes, not in this abstract type. All concrete types derived from this type shall extend this type to include one "...Value" element with an appropriate type, which should be one of the element types allowed in the ParameterValueType. In addition, all derived concrete types shall extend this type to include a "valueOfParameter" element that references one element substitutable for the "OperationParameter" element.

#### 12.6.4.2 **gml:parameterValue**

```
<element name="parameterValue" type="gml:ParameterValueType"
substitutionGroup="gml:_generalParameterValue"/>

<complexType name="ParameterValueType">
  <complexContent>
    <extension base="gml:AbstractGeneralParameterValueType">
      <sequence>
        <choice>
          <element ref="gml:value"/>
          <element ref="gml:dmsAngleValue"/>
          <element ref="gml:stringValue"/>
          <element ref="gml:integerValue"/>
          <element ref="gml:booleanValue"/>
          <element ref="gml:valueList"/>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

<element ref="gml:integerValueList"/>
<element ref="gml:valueFile"/>
</choice>
<element ref="gml:valueOfParameter"/>
</sequence>
</extension>
</complexContent>
</complexType>

```

A parameter value, ordered sequence of values, or reference to a file of parameter values. This concrete complexType can be used for operation methods without using an Application Schema that defines operation-method-specialized element names and contents, especially for methods with only one instance. This complexType can be used, extended, or restricted for well-known operation methods, especially for methods with many instances.

#### 12.6.4.3 gml:value

```
<element name="value" type="gml:MeasureType"/>
```

Numeric value of an operation parameter, with its associated unit of measure.

#### 12.6.4.4 gml:dmsAngleValue

```
<element name="dmsAngleValue" type="gml:DMSAngleType"/>
```

Value of an angle operation parameter, in either degree-minute-second format or single value format.

#### 12.6.4.5 gml:stringValue

```
<element name="stringValue" type="string"/>
```

String value of an operation parameter. A string value does not have an associated unit of measure.

#### 12.6.4.6 gml:integerValue

```
<element name="integerValue" type="positiveInteger"/>
```

Positive integer value of an operation parameter, usually used for a count. An integer value does not have an associated unit of measure.

#### 12.6.4.7 gml:booleanValue

```
<element name="booleanValue" type="boolean"/>
```

Boolean value of an operation parameter. A Boolean value does not have an associated unit of measure.

#### 12.6.4.8 gml:valueList

```
<element name="valueList" type="gml:MeasureListType"/>
```

Ordered sequence of two or more numeric values of an operation parameter list, where each value has the same associated unit of measure. An element of this type contains a space-separated sequence of double values.

#### 12.6.4.9 gml:integerValueList

```
<element name="integerValueList" type="gml:integerList"/>
```

Ordered sequence of two or more integer values of an operation parameter list, usually used for counts. These integer values do not have an associated unit of measure. An element of this type contains a space-separated sequence of integer values.

#### 12.6.4.10 **gml:valueFile**

```
<element name="valueFile" type="anyURI"/>
```

Reference to a file or a part of a file containing one or more parameter values, each numeric value with its associated unit of measure. When referencing a part of a file, that file shall contain multiple identified parts, such as an XML encoded document. Furthermore, the referenced file or part of a file can reference another part of the same or different files, as allowed in XML documents.

#### 12.6.4.11 **gml:valueOfParameter**

```
<element name="valueOfParameter" type="gml:OperationParameterRefType"/>
```

Association to the operation parameter that this is a value of.

#### 12.6.4.12 **gml:parameterValueGroup**

```
<element name="parameterValueGroup" type="gml:ParameterValueGroupType"
substitutionGroup="gml:_generalParameterValue"/>

<complexType name="ParameterValueGroupType">
  <complexContent>
    <extension base="gml:AbstractGeneralParameterValueType">
      <sequence>
        <element ref="gml:includesValue" minOccurs="2" maxOccurs="unbounded"/>
        <element ref="gml:valuesOfGroup"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A group of related parameter values. The same group can be repeated more than once in a Conversion, Transformation, or higher level parameterValueGroup, if those instances contain different values of one or more parameterValues which suitably distinguish among those groups. This concrete complexType can be used for operation methods without using an Application Schema that defines operation-method-specialized element names and contents. This complexType can be used, extended, or restricted for well-known operation methods, especially for methods with only one instance.

The includesValue elements are an unordered set of composition associations to the parameter values and groups of values included in this group.

#### 12.6.4.13 **gml:includesValue**

```
<element name="includesValue" type="gml:AbstractGeneralParameterValueType"
substitutionGroup="gml:_generalParameterValue"/>
```

A composition association to a parameter value or group of values included in this group.

#### 12.6.4.14 **gml:valuesOfGroup**

```
<element name="valuesOfGroup" type="gml:OperationParameterGroupRefType"/>
```

Association to the operation parameter group for which this element provides parameter values.

## 12.6.5 Operation method

### 12.6.5.1 gml:OperationMethod

```

<element name="OperationMethod" type="gml:OperationMethodType" substitutionGroup="gml:Definition"/>

<complexType name="OperationMethodType">
  <complexContent>
    <extension base="gml:OperationMethodBaseType">
      <sequence>
        <element ref="gml:methodID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:methodFormula"/>
        <element ref="gml:usesParameter" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="OperationMethodBaseType" abstract="true">
  <complexContent>
    <restriction base="gml:DefinitionType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:methodName"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>

```

Definition of an algorithm used to perform a coordinate operation. Most operation methods use a number of operation parameters, although some coordinate conversions use none. Each coordinate operation using the method assigns values to these parameters.

The methodID elements contain a set of alternative identifications of this operation method. The first methodID, if any, is normally the primary identification code, and any others are aliases.

The usesParameter elements are an unordered list of associations to the set of operation parameters and parameter groups used by this operation method.

The OperationMethodBaseType provides the basic encoding for operation method objects, simplifying and restricting the DefinitionType as needed.

The "remarks" element contains comments on or information about this reference system, including source information.

### 12.6.5.2 gml:methodName

```
<element name="methodName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which an operation method is identified.

### 12.6.5.3 gml:methodID

```
<element name="methodID" type="gml:IdentifierType"/>
```

An identification of this operation method.

### 12.6.5.4 gml:methodFormula

```
<element name="methodFormula" type="gml:CodeType"/>
```

Formula(s) used by this operation method. The value may be a reference to a publication. Note that the operation method may not be analytic, in which case this element references or contains the procedure, not an analytic formula.

#### 12.6.5.5 **gml:sourceDimensions**

```
<element name="sourceDimensions" type="positiveInteger"/>
```

Number of dimensions in the source CRS of this operation method.

#### 12.6.5.6 **gml:targetDimensions**

```
<element name="targetDimensions" type="positiveInteger"/>
```

Number of dimensions in the target CRS of this operation method

#### 12.6.5.7 **gml:usesParameter**

```
<element name="usesParameter" type="gml:AbstractGeneralOperationParameterRefType"/>
```

Association to an operation parameter or parameter group used by this operation method.

#### 12.6.5.8 **gml:operationMethodRef**

```
<element name="operationMethodRef" type="gml:OperationMethodRefType"
substitutionGroup="gml:dictionaryEntry"/>

<complexType name="OperationMethodRefType">
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:OperationMethod" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to a concrete general-purpose operation method, either referencing or containing the definition of that method.

### 12.6.6 Operation parameters and groups

#### 12.6.6.1 **gml:GeneralOperationParameter**

```
<element name="_GeneralOperationParameter" type="gml:AbstractGeneralOperationParameterType"
abstract="true" substitutionGroup="gml:Definition"/>

<complexType name="AbstractGeneralOperationParameterType" abstract="true">
  <complexContent>
    <extension base="gml:DefinitionType">
      <sequence>
        <element ref="gml:minimumOccurs" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Abstract definition of a parameter or group of parameters used by an operation method.

### 12.6.6.2 **gml:minimumOccurs**

```
<element name="minimumOccurs" type="nonNegativeInteger"/>
```

The minimum number of times that values for this parameter group or parameter are required. If this attribute is omitted, the minimum number is one.

### 12.6.6.3 **gml:abstractGeneralOperationParameterRef**

```
<element name="abstractGeneralOperationParameterRef" type="gml:AbstractGeneralOperationParameterRefType"
substitutionGroup="gml:dictionaryEntry"/>

<complexType name="AbstractGeneralOperationParameterRefType">
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:_GeneralOperationParameter" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to an operation parameter or group, either referencing or containing the definition of that parameter or group.

### 12.6.6.4 **gml:OperationParameter**

```
<element name="OperationParameter" type="gml:OperationParameterType"
substitutionGroup="gml:_GeneralOperationParameter"/>

<complexType name="OperationParameterType">
  <complexContent>
    <extension base="gml:OperationParameterBaseType">
      <sequence>
        <element ref="gml:parameterID" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="OperationParameterBaseType" abstract="true">
  <complexContent>
    <restriction base="gml:AbstractGeneralOperationParameterType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:parameterName"/>
        <element ref="gml:minimumOccurs" minOccurs="0"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
```

The definition of a parameter used by an operation method. Most parameter values are numeric, but other types of parameter values are possible. This complexType is expected to be used or extended for all operation methods, without defining operation-method-specialized element names.

The OperationParameterBaseType provides the basic encoding for operation parameter objects, simplifying and restricting the DefinitionType as needed.

The parameterID elements contain a set of alternative identifications of this operation parameter. The first parameterID, if any, is normally the primary identification code, and any others are aliases.

The "remarks" element contains comments on or information about this reference system, including source information.

#### 12.6.6.5 **gml:parameterName**

```
<element name="parameterName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which this operation parameter is identified.

#### 12.6.6.6 **gml:parameterID**

```
<element name="parameterID" type="gml:IdentifierType"/>
```

An identification of an operation parameter.

#### 12.6.6.7 **gml:operationParameterRef**

```
<element name="operationParameterRef" type="gml:OperationParameterRefType"/>

<complexType name="OperationParameterRefType">
  <complexContent>
    <restriction base="gml:AbstractGeneralOperationParameterRefType">
      <sequence>
        <element ref="gml:OperationParameter" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
```

Association to an operation parameter, either referencing or containing the definition of that parameter.

#### 12.6.6.8 **gml:OperationParameterGroup**

```
<element name="OperationParameterGroup" type="gml:OperationParameterGroupType"
  substitutionGroup="gml:_GeneralOperationParameter"/>

<complexType name="OperationParameterGroupType">
  <complexContent>
    <extension base="gml:OperationParameterGroup BaseType">
      <sequence>
        <element ref="gml:groupID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:maximumOccurs" minOccurs="0"/>
        <element ref="gml:includesParameter" minOccurs="2" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="OperationParameterGroup BaseType" abstract="true">
  <complexContent>
    <restriction base="gml:AbstractGeneralOperationParameterType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:groupName"/>
        <element ref="gml:minimumOccurs" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
```

```

        </restriction>
    </complexContent>
</complexType>
```

The definition of a group of parameters used by an operation method. This complexType is expected to be used or extended for all applicable operation methods, without defining operation-method-specialized element names.

The OperationParameterGroupBaseType provides the basic encoding for operation parameter group objects, simplifying and restricting the DefinitionType as needed.

The "groupID" elements contain a set of alternative identifications of this operation parameter group. The first groupID, if any, is normally the primary identification code, and any others are aliases.

The includesParameter elementsd are an unordered list of associations to the set of operation parameters that are members of this group.

The "remarks" element contains comments on or information about this reference system, including source information.

#### 12.6.6.9 **gml:groupName**

```
<element name="groupName" type="gml:SimpleNameType" substitutionGroup="gml:name"/>
```

The name by which this operation parameter group is identified.

#### 12.6.6.10 **gml:groupID**

```
<element name="groupID" type="gml:IdentifierType"/>
```

An identification of an operation parameter group.

#### 12.6.6.11 **gml:maximumOccurs**

```
<element name="maximumOccurs" type="positiveInteger"/>
```

The maximum number of times that values for this parameter group can be included. If this attribute is omitted, the maximum number is one.

#### 12.6.6.12 **gml:includesParameter**

```
<element name="includesParameter" type="gml:AbstractGeneralOperationParameterRefType"/>
```

Association to an operation parameter that is a member of a group.

#### 12.6.6.13 **gml:operationParameterGroupRef**

```

<element name="operationParameterGroupRef" type="gml:OperationParameterRefType"/>

<complexType name="OperationParameterGroupRefType">
    <complexContent>
        <restriction base="gml:AbstractGeneralOperationParameterRefType">
            <sequence>
                <element ref="gml:OperationParameterGroup" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
```

Association to an operation parameter, either referencing or containing the definition of that parameter.

## 12.7 dataQuality schema

### 12.7.1 Overview

Information about the accuracy or precision of coordinates, or of coordinate operations and coordinate operation parameters, is quality information and should be reported where possible in conformance with ISO 19115 and ISO 19114.

The parameters that define a coordinate reference system are chosen rather than measured to satisfy the degrees-of-freedom problem in the changeover from observation to coordinate quantities. Coordinate reference systems are therefore by definition error-free (i.e., non-stochastic). A coordinate reference system is realised through a network of control points. The coordinates of those control points, derived from surface and/or from satellite observations, are stochastic. Their accuracy can be expressed in a covariance matrix, which, due to the degrees-of-freedom problem, will have a rank deficiency, described in geodetic literature.

Coordinate transformations between coordinate reference systems usually have parameter values derived from two sets of point coordinates, one set in system 1, the other set in system 2. As these coordinates are stochastic (i.e., have random-error characteristics), the derived transformation parameter values will also be stochastic. Their covariance matrix can be calculated.

Coordinates that have not been ‘naturally’ determined in coordinate reference system 2, but have been determined in coordinate system 1 and then transformed to system 2, have the random error effects of the transformation superimposed on their original error characteristics. It may be possible in well-controlled cases to calculate the covariance matrices of the point coordinates before and after the transformation, and thus isolate the effect of the transformation, but in practice a user will only be interested in the accuracy of the final transformed coordinates.

Nevertheless the option is offered to specify the covariance matrix of point coordinates resulting exclusively from the transformation. It is outside the scope of this document to describe how that covariance matrix should be used. Because a covariance matrix is symmetrical, only the upper or lower diagonal part (including the main diagonal) needs to be specified.

For some transformations, this accuracy information is compacted in some assessment of an average impact on horizontal position and vertical position, allowing specification of average absolute accuracy and, when relevant and available, average relative accuracy. Hence separate quality measures may be specified for horizontal and for vertical position in those objects.

The data quality schema, dataQuality.xsd, specifies how to XML encode positional data quality information to describe the positional accuracy of coordinate operations. This schema encodes the Data Quality (DQ) package of the UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex B of Topic 2.

The dataQuality schema includes the units.xsd GML schema. This schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:dataQuality:v3.1.0

### 12.7.2 All elements

#### 12.7.2.1 gml:\_positionalAccuracy

```
<element name="_positionalAccuracy" type="gml:AbstractPositionalAccuracyType" abstract="true"/>
<complexType name="AbstractPositionalAccuracyType" abstract="true">
    <sequence>
```

```

<element ref="gml:measureDescription" minOccurs="0"/>
</sequence>
</complexType>

```

Position error estimate (or accuracy) data.

#### 12.7.2.2 gml:measureDescription

```
<element name="measureDescription" type="gml:CodeType"/>
```

A description of the position accuracy parameter(s) provided.

#### 12.7.2.3 gml:absoluteExternalPositionalAccuracy

```

<element name="absoluteExternalPositionalAccuracy" type="gml:AbsoluteExternalPositionalAccuracyType"
substitutionGroup="gml:_positionalAccuracy"/>

<complexType name="AbsoluteExternalPositionalAccuracyType">
<complexContent>
<extension base="gml:AbstractPositionalAccuracyType">
<sequence>
<element ref="gml:result"/>
</sequence>
</extension>
</complexContent>
</complexType>

```

Closeness of reported coordinate values to values accepted as or being true.

#### 12.7.2.4 gml:relativeInternalPositionalAccuracy

```

<element name="relativeInternalPositionalAccuracy" type="gml:RelativeInternalPositionalAccuracyType"
substitutionGroup="gml:_positionalAccuracy"/>

<complexType name="RelativeInternalPositionalAccuracyType">
<complexContent>
<extension base="gml:AbstractPositionalAccuracyType">
<sequence>
<element ref="gml:result"/>
</sequence>
</extension>
</complexContent>
</complexType>

```

Closeness of the relative positions of two or more positions to their respective relative positions accepted as or being true.

#### 12.7.2.5 gml:result

```
<element name="result" type="gml:MeasureType"/>
```

A quantitative result defined by the evaluation procedure used, and identified by the measureDescription.

#### 12.7.2.6 gml:covarianceMatrix

```

<element name="covarianceMatrix" type="gml:CovarianceMatrixType"
substitutionGroup="gml:_positionalAccuracy"/>

<complexType name="CovarianceMatrixType">
<complexContent>
<extension base="gml:AbstractPositionalAccuracyType">
<sequence>

```

```

        <element ref="gml:unitOfMeasure" maxOccurs="unbounded"/>
        <element ref="gml:includesElement" maxOccurs="unbounded"/>
    </sequence>
</extension>
</complexContent>
</complexType>
```

Error estimate covariance matrix.

The unitOfMeasure elements are an ordered sequence of units of measure, corresponding to the row and column index numbers of the covariance matrix, starting with row and column 1 and ending with row/column N. Each unit of measure is for the ordinate reflected in the relevant row and column of the covariance matrix.

The includesElement elements are an unordered set of elements in this covariance matrix. Because the covariance matrix is symmetrical, only the elements in upper or lower diagonal part (including the main diagonal) of the matrix need to be specified. Any zero valued covariance elements can be omitted

#### **12.7.2.7 gml:includesElement**

```

<element name="includesElement" type="gml:CovarianceElementType">

<complexType name="CovarianceElementType">
    <sequence>
        <element ref="gml:rowIndex"/>
        <element ref="gml:columnIndex"/>
        <element ref="gml:covariance"/>
    </sequence>
</complexType>
```

An element of a covariance matrix.

#### **12.7.2.8 gml:rowIndex**

```
<element name="rowIndex" type="positiveInteger"/>
```

Row number of this covariance element value.

#### **12.7.2.9 gml:columnIndex**

```
<element name="columnIndex" type="positiveInteger"/>
```

Column number of this covariance element value.

#### **12.7.2.10 gml:covariance**

```
<element name="covariance" type="double"/>
```

Value of covariance matrix element.

## **13 GML schemas – topology**

### **13.1 Introduction**

Topology is the branch of mathematics describing the properties of objects which are invariant under continuous deformation. For example, a circle is topologically equivalent to an ellipse because one can be transformed into the other by stretching. In geographic modelling, the foremost use of topology is in accelerating computational geometry. The constructs of topology allow characterisation of the spatial relationships between objects using simple combinatorial or algebraic algorithms. Topology, realised by the

appropriate geometry, also allows a compact and unambiguous mechanism for expressing shared geometry among geographic features.

## 13.2 Topology Model

The conceptual model underlying the representation of topology in GML is that of Topic 1 of the OGC Abstract Specification (ISO 19107). The model describes the correspondence of topological and geometric relationships up to 3 dimensions – volume topology. Relevant parts of the model are summarised here for convenience.

There are four instantiable classes of primitive topology objects, one for each dimension up to 3D. In addition, topological complexes are supported, too.

There is strong symmetry in the (topological boundary and coboundary) relationships between topology primitives of adjacent dimensions. Topology primitives are bounded by directed primitives of one lower dimension. The coboundary of each topology primitive is formed from directed topology primitives of one higher dimension.

## 13.3 Types and elements

### 13.3.1 Schema Includes

```
<include schemaLocation="geometryComplexes.xsd"/>
```

The topology schema includes definitions from the geometry schema yielding the ability to describe primitives and complexes with a geometric realisation.

### 13.3.2 `gml:AbstractTopologyType`, `gml:_Topology`

```
<complexType name="AbstractTopologyType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractGMLType"/>
  </complexContent>
</complexType>

<element name="_Topology" type="gml:AbstractTopologyType" abstract="true" substitutionGroup="gml:_Object"/>
```

This abstract type supplies the root or base type for all topological elements including primitives and complexes. It inherits AbstractGMLType and hence can be identified using the gml:id attribute.

### 13.3.3 `gml:AbstractTopoPrimitive`, `gml:_TopoPrimitive`

```
<complexType name="AbstractTopoPrimitiveType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractTopologyType">
      <sequence>
        <element ref="gml:isolated" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:container" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="_TopoPrimitive" type="gml:AbstractTopoPrimitiveType" abstract="true"
  substitutionGroup="gml:_Topology"/>
```

This abstract type acts as the base type for all topological primitive elements. Topology primitives are the atomic (smallest possible) units of a topology complex. Each topology primitive may contain references to other topology primitives of codimension 2 or more. So faces may isolate nodes and TopoSolids may isolate

nodes and edges. Conversely, nodes may have faces as containers and nodes and edges may have TopoSolids as containers.

#### 13.3.4 gml:NodeType, gml:Node

```
<complexType name="NodeType">
<complexContent>
<extension base="gml:AbstractTopoPrimitiveType">
<sequence>
<element ref="gml:directedEdge" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:pointProperty" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>

<element name="Node" type="gml:NodeType" substitutionGroup="gml:_TopoPrimitive"/>
```

The Node type and global element represent the 0-dimensional primitive expressing point coincidence. The topological boundary of a node is empty and hence requires no representation. The optional coboundary of a node is a set of directed edges which are incident on this node.

**NOTE** In a 2D complex, this set may be ordered as a clockwise circular sequence. In a 3D complex, the order of this set is arbitrary.

Edges emanating from this node appear in the node coboundary with a negative orientation.

A node may optionally be realised by a 0-dimensional (point) geometric primitive.

#### 13.3.5 gml:DirectedNode.PropertyType, gml:directedNode

```
<complexType name="DirectedNode.PropertyType">
<choice>
<element ref="gml:Node" minOccurs="0"/>
</choice>
<attribute name="orientation" type="gml:SignType" default="+"/>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="directedNode" type="gml:DirectedNode.PropertyType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:directedNode">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
</annotation>
</element>
```

The role of the DirectedNode type and element is in the boundary of topology edges and in the support of topological point features via the TopoPoint expression, see below. The orientation attribute of type gml:SignType expresses the sense in which the included node is used e.g. start (“-”) or end (“+”) node.

#### 13.3.6 gml:EdgeType, gml:Edge

```
<complexType name="EdgeType">
<complexContent>
<extension base="gml:AbstractTopoPrimitiveType">
<sequence>
<element ref="gml:directedNode" minOccurs="2" maxOccurs="2"/>
```

```

<element ref="gml:directedFace" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:curveProperty" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>

<element name="Edge" type="gml:EdgeType" substitutionGroup="gml:_TopoPrimitive"/>

```

The Edge type and global element represent the 1-dimensional primitive expressing linear coincidence. The topological boundary of an Edge consists of a negatively directed start Node and a positively directed end Node. The optional coboundary of an edge is a circular sequence of directed faces which are incident on this edge in document order. Faces which use a particular boundary edge in its positive orientation appear with positive orientation on the coboundary of the same edge. In the 2D case, the orientation of the face on the left of the edge is "+"; the orientation of the face on the right on its right is "-". An edge may optionally be realised by a 1-dimensional (curve) geometric primitive.

### 13.3.7 gml:DirectedEdgePropertyType, gml:directedEdge

```

<complexType name="DirectedEdgePropertyType">
<choice>
<element ref="gml:Edge" minOccurs="0"/>
</choice>
<attribute name="orientation" type="gml:SignType" default="+"/>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="directedEdge" type="gml:DirectedEdgePropertyType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:directedEdge">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
</annotation>
</element>

```

The role of the DirectedEdge type and element is in the boundary of topology faces, in the coBoundary of topology nodes and in the support of topological line features via the TopoCurve expression, see below. The orientation attribute of type gml:SignType expresses the sense in which the included edge is used e.g. forward or reverse.

### 13.3.8 gml:FaceType, gml:Face

```

<complexType name="FaceType">
<complexContent>
<extension base="gml:AbstractTopoPrimitiveType">
<sequence>
<element ref="gml:directedEdge" maxOccurs="unbounded"/>
<element ref="gml:directedTopoSolid" minOccurs="0" maxOccurs="2"/>
<element ref="gml:surfaceProperty" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>

<element name="Face" type="gml:FaceType" substitutionGroup="gml:_TopoPrimitive"/>

```

The Face type and global element represent the 2-dimensional topology primitive expressing surface overlap. The topological boundary of a face consists of a set of directed edges. Note that all edges associated with the face, including dangling edges, appear in the boundary. A dangling edge has the same face on both sides. Consequently, a dangling edge has two different nodes in its boundary. A dangling edge may share zero, one

or two bounding nodes with other edges in the boundary of a face. Two directedEdge elements with opposite orientations reference each dangling edge in the boundary of a face. The non-dangling edges in the boundary of a face comprise one or more topological rings. Each such ring consists of directedEdges connected in a cycle, and is oriented with the face on its left. The optional coboundary of a face is a pair of directed solids which are bounded by this face. If present, there is precisely one positively directed and one negatively directed solid in the coboundary of every face. The positively directed solid corresponds to the solid which lies in the direction of the negatively directed normal to the face in any geometric realisation. A face may optionally be realised by a 2-dimensional (surface) geometric primitive.

### 13.3.9 gml:DirectedFacePropertyType, gml:directedFace

```

<complexType name="DirectedFacePropertyType">
  <choice>
    <element ref="gml:Face" minOccurs="0"/>
  </choice>
  <attribute name="orientation" type="gml:SignType" default="+"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="directedFace" type="gml:DirectedFacePropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:directedFace">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
  </annotation>
</element>
```

The role of the DirectedFace type and element is in the boundary of topology solids, in the coBoundary of topology edges and in the support of surface features via the TopoSurface expression, see below. The orientation attribute of type gml:SignType expresses the sense in which the included face is used e.g. inward or outward with respect to the surface normal in any geometric realisation.

### 13.3.10 gml:TopoSolidType, gml:TopoSolid

```

<complexType name="TopoSolidType">
  <complexContent>
    <extension base="gml:AbstractTopoPrimitiveType">
      <sequence>
        <element ref="gml:directedFace" maxOccurs="unbounded"/>
        <element ref="gml:solidProperty" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="TopoSolid" type="gml:TopoSolidType" substitutionGroup="gml:_TopoPrimitive"/>
```

The TopoSolid type and global element represent the 3-dimensional topology primitive expressing Volume interclause. The topological boundary of a TopoSolid consists of a set of directed faces. Note that all faces associated with the TopoSolid, including dangling faces, appear in the boundary. A dangling face has the same solid on both sides. Two directedFace elements with opposite orientations reference each dangling face in the boundary of a topological solid. The coboundary of a TopoSolid is empty and hence requires no representation. A TopoSolid may optionally be realised by a 2-dimensional (solid) geometric primitive.

### 13.3.11 gml:DirectedTopoSolidPropertyType, gml:directedTopoSolid

```
<complexType name="DirectedTopoSolidPropertyType">
```

```

<choice>
  <element ref="gml:TopoSolid"/>
</choice>
<attribute name="orientation" type="gml:SignType" default="+"/>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="directedTopoSolid" type="gml:DirectedTopoSolidPropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:directedTopoSolid">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
  </annotation>
</element>

```

The role of the DirectedSolid type and element is in the coBoundary of topology faces and in the support of 3D volume features via the TopoVolume expression, see below. The orientation attribute of type gml:SignType expresses the sense in which the included solid appears in the face coboundary. Note that in the context of a TopoVolume the orientation attribute has no meaning in 3D.

### 13.3.12 gml:isolated

```

<element name="isolated" type="gml:IsolatedPropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:isolated">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
  </annotation>
</element>

<complexType name="IsolatedPropertyType">
  <choice minOccurs="0">
    <element ref="gml:Node"/>
    <element ref="gml:Edge"/>
  </choice>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

```

All of the adjacency relationships between topology primitives whose dimensions differ by +/- 1 are described using boundary and coboundary through directed topology primitive properties as described for each primitive. This includes instances where a dangling edge has the same face on both sides or a dangling face has the same solid on both sides. These primitives will appear twice in the boundary of the relevant face or solid with alternate sign. Primitives which are enclosed by another primitive of at least codimension 2 however, with no intermediate dimension primitive, are truly isolated. For faces this corresponds to nodes with no relationship to a bounding edge of the face. For solids, it corresponds to nodes or edges which are not referred to by the boundary faces of the solid.

Note a node may be isolated in a face and yet appear in the boundary of an edge which does *not* form part of the boundary of the face. In 3D this corresponds geometrically to a curve whose endpoint meets the interior of a surface.

### 13.3.13 gml:container

```

<element name="container" type="gml:ContainerPropertyType">
  <annotation>

```

```

<appinfo>
  <sch:pattern>
    <sch:rule context="gml:containerProperty">
      <sch:extends rule="hrefOrContent"/>
    </sch:rule>
  </sch:pattern>
</appinfo>
</annotation>
</element>

<complexType name="Container.PropertyType">
  <choice minOccurs="0">
    <element ref="gml:Face"/>
    <element ref="gml:TopoSolid"/>
  </choice>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

```

The container type and element optionally contain the reciprocal relationships for the isolated property described above.

### 13.3.14 gml:TopoPointType, gml:TopoPoint

```

<complexType name="TopoPointType">
  <complexContent>
    <extension base="gml:AbstractTopologyType">
      <sequence>
        <element ref="gml:directedNode"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="TopoPoint" type="gml:TopoPointType"/>

```

The intended use of TopoPoint is to appear within a point feature to express the structural and possibly geometric relationships of this point to other features via shared node definitions. Note the orientation assigned to the directedNode has no meaning in this context. It is preserved for symmetry with the types and elements which follow.

### 13.3.15 gml:TopoPointPropertyType, gml:topoPointProperty

```

<complexType name="TopoPointPropertyType">
  <sequence>
    <element ref="gml:TopoPoint"/>
  </sequence>
</complexType>

<element name="topoPointProperty" type="gml:TopoPointPropertyType"/>

```

The topoPointProperty element can be used as a property of features to express their relationship to the referenced topology node.

### 13.3.16 gml:TopoCurveType, gml:TopoCurve

```

<complexType name="TopoCurveType">
  <complexContent>
    <extension base="gml:AbstractTopologyType">
      <sequence>
        <element ref="gml:directedEdge" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

</complexContent>
</complexType>

<element name="TopoCurve" type="gml:TopoCurveType"/>
```

The TopoCurve type and element represent a homogeneous topological expression, a list of directed edges, which if realised are isomorphic to a geometric curve primitive. The intended use of TopoCurve is to appear within a line feature instance to express the structural and geometric relationships of this line to other features via the shared edge definitions.

### 13.3.17 gml:TopoCurvePropertyType, gml:topoCurveProperty

```

<complexType name="TopoCurvePropertyType">
  <sequence>
    <element ref="gml:TopoCurve"/>
  </sequence>
</complexType>

<element name="topoCurveProperty" type="gml:TopoCurvePropertyType"/>
```

The topoCurveProperty element can be used as a property of features to express their relationship to the referenced topology edges.

### 13.3.18 gml:TopoSurfaceType, gml:TopoSurface

```

<complexType name="TopoSurfaceType">
  <complexContent>
    <extension base="gml:AbstractTopologyType">
      <sequence>
        <element ref="gml:directedFace" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="TopoSurface" type="gml:TopoSurfaceType"/>
```

The TopoSurface type and element represent a homogeneous topological expression, a set of directed faces, which if realised are isomorphic to a geometric surface primitive. The intended use of TopoSurface is to appear within a surface feature instance to express the structural and possibly geometric relationships of this surface to other features via the shared face definitions.

### 13.3.19 gml:TopoSurfacePropertyType, gml:topoSurfaceProperty

```

<complexType name="TopoSurfacePropertyType">
  <sequence>
    <element ref="gml:TopoSurface"/>
  </sequence>
</complexType>

<element name="topoSurfaceProperty" type="gml:TopoSurfacePropertyType"/>
```

The topoSurfaceProperty element can be used as a property of features to express their relationship to the referenced topology faces.

### 13.3.20 gml:TopoVolumeType, gml:TopoVolume

```

<complexType name="TopoVolumeType">
  <complexContent>
    <extension base="gml:AbstractTopologyType">
      <sequence>
        <element ref="gml:directedTopoSolid" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

        </sequence>
    </extension>
</complexContent>
</complexType>

<element name="TopoVolume" type="gml:TopoVolumeType"/>

```

The TopoVolume type and element represent a homogeneous topological expression, a set of directed TopoSolids, which if realised are isomorphic to a geometric solid primitive. The intended use of TopoVolume is to appear within a 3D solid feature instance to express the structural and geometric relationships of this solid to other features via the shared TopoSolid definitions. Note the orientation assigned to the directedSolid has no meaning in three dimensions. It is preserved for symmetry with the preceding types and elements.

### 13.3.21 gml:TopoVolumePropertyType, gml:topoVolumeProperty

```

<complexType name="TopoVolumePropertyType">
    <sequence>
        <element ref="gml:TopoVolume"/>
    </sequence>
</complexType>

<element name="topoVolumeProperty" type="gml:TopoVolumePropertyType"/>

```

The topoVolumeProperty element can be used as a property of features to express their relationship to the referenced topology Volume.

### 13.3.22 gml:TopoComplexType, gml:TopoComplex

A TopoComplex in GML is a collection of disconnected TopoPrimitives. If a topological primitive (other than a Node) is in a particular TopoComplex, then there exists a set of primitives of lower dimension in the same complex that form the boundary of this primitive.

```

<complexType name="TopoComplexType">
    <complexContent>
        <extension base="gml:AbstractTopologyType">
            <sequence>
                <element ref="gml:maximalComplex"/>
                <element ref="gml:superComplex" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:subComplex" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:topoPrimitiveMember" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:topoPrimitiveMembers" minOccurs="0"/>
            </sequence>
            <attribute name="isMaximal" type="boolean" default="false"/>
        </extension>
    </complexContent>
</complexType>

<element name="TopoComplex" type="gml:TopoComplexType" substitutionGroup="gml:_Topology"/>

```

This type and element provide encoding for a topology complex comprising multiple topology primitive members. In addition to primitives, each complex holds a reference to a unique maximal complex (the complex which has no supercomplex) and optionally to some number of sub- or super- complexes. A topology complex contains its primitive and sub-complex members, and is contained by its super-complex(es). The primitive and sub-complex members of a topological complex have dimensionality less than or equal to the dimensionality of the topology complex. There is one and only one maximal complex per topological manifold.

### 13.3.23 Maximal, sub- and super-complexes

```

<complexType name="TopoComplexMemberType">
    <sequence>
        <element ref="gml:TopoComplex" minOccurs="0"/>

```

```

</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="subComplex" type="gml:TopoComplexMemberType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:subComplex">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
</annotation>
</element>

<element name="superComplex" type="gml:TopoComplexMemberType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:superComplex">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
</annotation>
</element>

<element name="maximalComplex" type="gml:TopoComplexMemberType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:subComplex">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
<documentation>Need schamatron test here that isMaximal attribute value is true</documentation>
</annotation>
</element>

```

These elements and type provide encoding for relationships between topology complexes as described for the TopoComplexType.

### 13.3.24 gml:topoPrimitiveMember

```

<element name="topoPrimitiveMember" type="gml:topoPrimitiveMemberType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:topoPrimitiveMember">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
</annotation>
</element>

<complexType name="topoPrimitiveMemberType">
<sequence>
<element ref="gml:_TopoPrimitive" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

```

This type and element encode the relationship between a topology complex and a single topology primitive.

### 13.3.25 **gml:topoPrimitiveMembers**

```
<element name="topoPrimitiveMembers" type="gml:TopoPrimitiveArrayAssociationType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:topoPrimitiveMember">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
</annotation>
</element>

<complexType name="TopoPrimitiveArrayAssociationType">
<complexContent>
<restriction base="gml:ArrayAssociationType">
<sequence>
<element ref="gml:_TopoPrimitive" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

This type and element encode the relationship between a topology complex and an arbitrary number of topology primitives. Note that because of the array style of encoding this type requires that the encodings for the topology primitives be in-line rather than remote properties.

### 13.3.26 **gml:TopoComplexProperty, gml:topoComplexProperty**

```
<complexType name="TopoComplexMemberType">
<sequence>
<element ref="gml:TopoComplex" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="topoComplexProperty" type="gml:TopoComplexMemberType"/>
```

This type and element encode the relationship between a GML object such as a feature collection and a topological complex. They allow a feature collection to contain or reference a topological complex that contains topologies referenced by members of the feature collection.

## 14 GML schemas – temporal information and dynamic features

### 14.1 Overview

The GML temporal schemas include components for describing temporal geometry and topology, temporal reference systems, and the temporal characteristics of geographic data. The model underlying the representation constitutes a profile of the conceptual schema described in ISO 19108:2002 [i]. The underlying spatiotemporal model strives to accommodate both feature-level and attribute-level time stamping; basic support for tracking moving objects is also included.

Time is measured on two types of scales: interval and ordinal. An interval scale offers a basis for measuring duration; an ordinal scale provides information only about relative position in time (e.g. a stratigraphic sequence or the geological time scale).

Two other ISO standards are relevant to describing temporal objects: ISO 8601 describes encodings for time instants and time periods, as text strings with particular structure and punctuation; ISO 11404 provides a detailed description of time intervals as part of a general discussion of language independent datatypes.

The temporal schemas cover two interrelated topics: three schema documents provide basic elements for representing temporal instants and periods, temporal topology, and reference systems; a more specialized schema document defines components used for dynamic features. Instances of temporal geometric types are used as values for the temporal properties of geographic features. All schemas are listed in Annex C.

## 14.2 Temporal schema

### 14.2.1 Overview

The basic temporal types and elements are described in the schema listed in Annex C. The main temporal schema is identified by the following location-independent name (using URN syntax):

### 14.2.2 urn:opengis:specification:gml:schema-xsd:temporal:v3.1.0Temporal objects and properties

A hierarchy of elements is provided that matches the conceptual schema described in ISO 19108:2002. This includes a number of abstract (non-instantiable) elements at the top of the hierarchy.

#### 14.2.2.1 gml:\_TimeObject

The abstract element gml:\_TimeObject acts as the head of a substitution group for temporal primitives and complexes. It is declared in the schema as follows:

```
<element name="_TimeObject" type="gml:AbstractTimeObjectType" abstract="true" substitutionGroup="gml:_GML"/>
```

A gml:\_TimeObject may be used in any position that a gml:\_ GML is valid. Its content model is defined as follows:

```
<complexType name="AbstractTimeObjectType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractGMLType"/>
  </complexContent>
</complexType>
```

**NOTE** The content model of gml:\_TimeObject is a vacuous extension of AbstractGMLType. Types derived from this have the standard GML object properties available: metaDataProperty, description, name.

#### 14.2.2.2 gml:\_TimePrimitive

The abstract element TimePrimitive acts as the head of a substitution group for geometric and topological temporal primitives. It is declared in the schema as follows:

```
<element name="_TimePrimitive" type="gml:AbstractTimePrimitiveType" abstract="true"
substitutionGroup="gml:_TimeObject"/>
```

A gml:\_TimePrimitive may be used in any position that a gml:\_ TimeObject is valid. Its content model is defined as follows:

```
<complexType name="AbstractTimePrimitiveType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractTimeObjectType">
      <sequence>
        <element name="relatedTime" type="gml:RelatedTimeType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

This extends the model for generic temporal objects with properties indicating relationships between this temporal primitive and other temporal primitives. The definition of `gml:RelatedTimeType` is provided in subclause 14.2.2.4 below.

#### 14.2.2.3 `gml:TimePrimitivePropertyType`, `gml:validTime`

`gml:TimePrimitivePropertyType` provides a standard content model for associations between an arbitrary member of the substitution group whose head is `gml:_TimePrimitive` and another object:

```
<complexType name="TimePrimitivePropertyType">
  <sequence minOccurs="0">
    <element ref="gml:_TimePrimitive"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

`gml:validTime` is a convenience element declared as follows:

```
<element name="validTime" type="gml:TimePrimitivePropertyType"/>
```

#### 14.2.2.4 `gml:RelatedTimeType`

`gml:RelatedTimeType` provides a content model for indicating the relative position of an arbitrary member of the substitution group whose head is `gml:_TimePrimitive`. It extends the generic `gml:TimePrimitivePropertyType` with an XML attribute “`relativePosition`”, whose value is selected from the set of 13 temporal relationships identified by Allen (1983) [ISO 19108:2002]:

```
<complexType name="RelatedTimeType">
  <complexContent>
    <extension base="gml:TimePrimitivePropertyType">
      <attribute name="relativePosition">
        <simpleType>
          <restriction base="string">
            <enumeration value="Before"/>
            <enumeration value="After"/>
            <enumeration value="Begins"/>
            <enumeration value="Ends"/>
            <enumeration value="During"/>
            <enumeration value="Equals"/>
            <enumeration value="Contains"/>
            <enumeration value="Overlaps"/>
            <enumeration value="Meets"/>
            <enumeration value="OverlappedBy"/>
            <enumeration value="MetBy"/>
            <enumeration value="BegunBy"/>
            <enumeration value="EndedBy"/>
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </complexContent>
</complexType>
```

#### 14.2.2.5 `gml:_TimeComplex`

A temporal complex shall be an aggregation of temporal primitives, represents a temporal geometric complex and a temporal topology complex. Temporal geometric complex is not defined in this clause. This standard does not distinguish a temporal linear graph from a temporal non-linear graph.

The abstract element TimeComplex acts as the head of a substitution group for temporal complexes. It is declared in the schema as follows:

```
<element name="_TimeComplex" type="gml:AbstractTimeComplexType" abstract="true"
substitutionGroup="gml:_TimeObject"/>
```

A gml:\_TimeComplex may be used in any position that a gml:\_TimeObject is valid. Its content model is defined as follows:

```
<complexType name="AbstractTimeComplexType" abstract="true">
<complexContent>
<extension base="gml:AbstractTimeObjectType"/>
</complexContent>
</complexType>
```

#### 14.2.3 Temporal geometry

Temporal geometry is described in terms of time instants, periods, positions and lengths.

##### 14.2.3.1 gml:\_TimeGeometricPrimitive

The abstract element TimeGeometricPrimitive acts as the head of a substitution group for geometric temporal primitives. It is declared in the schema as follows:

```
<element name="_TimeGeometricPrimitive" type="gml:AbstractTimeGeometricPrimitiveType" abstract="true"
substitutionGroup="gml:_TimePrimitive"/>
```

A gml:\_TimeGeometricPrimitive may be used in any position that a gml:\_TimePrimitive is valid. Its content model is defined as follows:

```
<complexType name="AbstractTimeGeometricPrimitiveType" abstract="true">
<complexContent>
<extension base="gml:AbstractTimePrimitiveType">
<attribute name="frame" type="anyURI" use="optional" default="#ISO-8601"/>
</extension>
</complexContent>
</complexType>
```

A temporal geometry shall be associated with a temporal reference system through the **frame** attribute that provides a URI reference that identifies a description of the reference system. Following ISO 8601, the Gregorian calendar with UTC is the default reference system, but others may also be used. The GPS calendar and the Julian calendar are alternative reference systems in common use.

The two geometric primitives in the temporal dimension are the instant and the period. GML components are defined to support these as follows.

##### 14.2.3.2 gml:TimeInstant

An instant is a zero-dimensional geometric primitive that represents an identifiable position in time (it is equivalent to a point in space). In practice, an instant is an interval whose duration is less than a chronon—the resolution of the time scale..

The element **gml:TimeInstant** is declared as follows:

```
<element name="TimeInstant" type="gml:TimeInstantType" substitutionGroup="gml:_TimeGeometricPrimitive"/>
```

A gml:TimeInstant may be used in any position that a gml:\_TimeGeometricPrimitive is valid. Its content model is defined as follows:

```
<complexType name="TimeInstantType" final="#all">
<complexContent>
```

```

<extension base="gml:AbstractTimeGeometricPrimitiveType">
  <sequence>
    <element ref="gml:timePosition"/>
  </sequence>
</extension>
</complexContent>
</complexType>

```

In an instance document, a gml:TimeInstant contains a gml:timePosition as follows:

```

<gml:TimeInstant gml:id="t11">
  <gml:description>Abby's birthday</gml:description>
  <gml:timePosition>2001-05-23</gml:timePosition>
</gml:TimeInstant>

```

#### 14.2.3.3 gml:TimeInstantPropertyType

gml:TimeInstantPropertyType is a specialisation of gml:TimePrimitivePropertyType that provides for associating a gml:TimeInstant with an object:

```

<complexType name="TimeInstantPropertyType">
  <sequence minOccurs="0">
    <element ref="gml:TimeInstant"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

```

#### 14.2.3.4 gml:TimePeriod

A period is a one-dimensional geometric primitive that represents an identifiable extent in time (it is equivalent to a curve or envelope in space). It is an open interval bounded by beginning and end points (i.e. instants), and has length (i.e. duration). Its location in time is described by the temporal positions of the instants at which it begins and ends. The length of the period is equal to the temporal distance between the two bounding temporal positions.

The element **gml:TimePeriod** is declared as follows:

```
<element name="TimePeriod" type="gml:TimePeriodType" substitutionGroup="gml:_TimeGeometricPrimitive"/>
```

gml:TimePeriod may be used in any position that a gml:\_TimeGeometricPrimitive is valid. Its content model is defined as follows:

```

<complexType name="TimePeriodType">
  <complexContent>
    <extension base="gml:AbstractTimeGeometricPrimitiveType">
      <sequence>
        <choice>
          <element name="beginPosition" type="gml:TimePositionType"/>
          <element name="begin" type="gml:TimeInstantPropertyType"/>
        </choice>
        <choice>
          <element name="endPosition" type="gml:TimePositionType"/>
          <element name="end" type="gml:TimeInstantPropertyType"/>
        </choice>
        <element ref="gml:_timeLength" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

Both beginning and end may be described in terms of their direct position using `gml:TimePositionType` (clause 0) which is an XML Schema simpleContent type, or by reference to an identifiable time instant using `gml:TimeInstantPropertyType` (clause 14.2.3.3).

For example, within a `gml:TimePeriod`, a `gml:TimeInstant` may appear directly as the value of the begin and end as follows:

```
<gml:TimePeriod gml:id="p22">
  <gml:begin>
    <gml:TimeInstant gml:id="t11">
      <gml:timePosition>2001-05-23</gml:timePosition>
    </gml:TimeInstant>
  </gml:begin>
  <gml:end>
    <gml:TimeInstant gml:id="t12">
      <gml:timePosition>2001-06-23</gml:timePosition>
    </gml:TimeInstant>
  </gml:end>
</gml:TimePeriod>
```

Alternatively a limit of a `gml:timePeriod` may use the conventional GML property model to make a reference to a time instant described elsewhere, or a limit may be indicated as a direct position. The following mixed example shows both of these, as well as including the optional `gml:duration` property:

```
<gml:TimePeriod gml:id="p22">
  <gml:begin xlink:href="#t11"/>
  <gml:endPosition>2002-05-23</gml:endPosition>
  <gml:duration>P1Y</gml:duration>
</gml:TimePeriod>
```

#### 14.2.3.5 `gml:timePeriodPropertyType`

`gml:timePeriodPropertyType` is a specialisation of `gml:TimePrimitivePropertyType` that provides for associating a `gml:TimePeriod` with an object:

```
<complexType name="TimePeriodPropertyType">
  <sequence minOccurs="0">
    <element ref="gml:TimePeriod"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

#### 14.2.3.6 `gml:TimePositionType`, `gml:timePosition`

The method for identifying a temporal position is specific to each temporal reference system. `gml:timePositionType` supports the description of temporal position according to the subtypes described in ISO 19108:2002. Values based on calendars and clocks use lexical formats that are based on the ISO 8601 standard, as described in Part 2 of the XML Schema recommendation. A decimal value may be used with coordinate systems such as GPS time or UNIX time. A URI may be used to provide a reference to some era in an ordinal reference system (e.g. a geological epoch).

In common with many of the components using the stereotype <<DataType>> in the ISO 19100 specifications, the corresponding GML component has simpleContent. However, the content model `gml:TimePositionType` is defined in several steps:

```
<complexType name="TimePositionType" final="#all">
  <simpleContent>
    <extension base="gml:TimePositionUnion">
      <attribute name="frame" type="anyURI" use="optional" default="#ISO-8601"/>
      <attribute name="calendarEraName" type="string" use="optional"/>
      <attribute name="indeterminatePosition" type="gml:TimeIndeterminateValueType" use="optional"/>
    </extension>
```

```
</simpleContent>
</complexType>
```

Three XML attributes appear on `gml:TimePositionType`:

A time value shall be associated with a temporal reference system through the **frame** attribute that provides a URI reference that identifies a description of the reference system. Following ISO 8601, the Gregorian calendar with UTC is the default reference system, but others may also be used. The GPS calendar and the Julian calendar are alternative reference systems in common use. Components for describing temporal reference systems are described in clause 14.3, but it is not required that the reference system be described in this, as the reference can refer to anything that may be identified with a URI.

For time values using a calendar containing more than one era, the (optional) **calendarEraName** attribute provides the name of the calendar era.

Inexact temporal positions may be expressed using the optional **indeterminatePosition** attribute. This takes a value from an enumeration defined as follows:

```
<simpleType name="TimeIndeterminateValueType">
  <restriction base="string">
    <enumeration value="after"/>
    <enumeration value="before"/>
    <enumeration value="now"/>
    <enumeration value="unknown"/>
  </restriction>
</simpleType>
```

These values are interpreted as follows:

- “unknown” indicates that no specific value for temporal position is provided.
- “now” indicates that the specified value shall be replaced with the current temporal position whenever the value is accessed.
- “before” indicates that the actual temporal position is unknown, but it is known to be before the specified value.
- “after” indicates that the actual temporal position is unknown, but it is known to be after the specified value.

A value for `indeterminatePosition` can be used either

- alone, or
- can qualify a specific value for temporal position (e.g. before 2002-12, after 1019624400).

The simpleType `gml:TimePositionUnion` is a union of XML Schema simple types which instantiate the subtypes for temporal position described in ISO 19108.

```
<simpleType name="TimePositionUnion">
  <union memberTypes="gml:CalDate time dateTime anyURI decimal"/>
</simpleType>
```

An ordinal era may be referenced via URI. A decimal value can be used to indicate the distance from the scale origin (e.g. UNIX time, GPS calendar). **time** is used for a position that recurs daily (see clause 5.4.4.2 of ISO 19108:2002).

Finally, calendar and clock forms that support the representation of time in systems based on years, months, days, hours, minutes and seconds, in a notation following ISO 8601, are assembled as follows:

```
<simpleType name="CalDate">
  <union memberTypes="date gYearMonth gYear"/>
</simpleType>
```

NOTE The XML Schema simpleType **dateTime** does not permit right-truncation, except for fractions of seconds, which is why **date**, **gYear** and **gYearMonth** are required.

NOTE Following ISO 19108:2002, when used with non-Gregorian calendars based on years, months, days, the same lexical representation should still be used. Following XML Schema part 2, leading zeros should be added if the year value would otherwise have fewer than four digits.

The element **gml:timePosition** is declared as follows:

```
<element name="timePosition" type="gml:TimePositionType"/>
```

This element is used directly as a property of **gml:TimeInstant** (see clause 14.2.3.2), and may also be used in application schemas. The following examples illustrate how **gml:timePosition** or other elements of this type may appear in a data instance:

```
<gml:timePosition>2002-11-25T13:20:20</gml:timePosition>
<gml:timePosition indeterminatePosition="after">1994</gml:timePosition>
<gml:timePosition indeterminatePosition="now"></gml:timePosition>
<gml:timePosition frame="http://my.big.org/TRS/GPS">25876321.01</gml:timePosition>
<gml:timePosition frame="http://my.big.org/TRS/archaeology"> http://my.history.org/eras/bronzeAge </gml:timePosition>
<gml:timePosition frame="http://my.big.org/TRS/calendars/japanese" calendarEraName="Meiji">0085-03</gml:timePosition>
```

#### 14.2.3.7 **gml:TimeLengthType**, **gml:\_timeLength**, **gml:duration**, **gml:timeInterval**, **gml:TimeUnitType**

The length of a time period is described using the element **gml:\_timeLength**, which is declared in the schema as follows:

```
<element name="_timeLength" type="gml:TimeDurationType" abstract="true">
```

**gml:\_timeLength** is an abstract element, which acts as the head of a substitution group. Its content model is a union of the XML Schema duration and decimal simpleTypes, defined as follows:

```
<simpleType name="TimeLengthType">
  <union memberTypes="duration decimal"/>
</simpleType>
```

One member of the substitution group is **gml:duration**, which conforms to the ISO 8601 syntax for temporal length as implemented by the XML Schema duration type:

```
<element name="duration" type="duration" substitutionGroup="gml:_timeLength"/>
```

Another member of the substitution group is **gml:timeInterval** which conforms to the ISO 11404 standard which is based on floating point values for temporal length.

```
<element name="timeInterval" type="gml:TimeIntervalLengthType" substitutionGroup="gml:_timeLength"/>

<complexType name="TimeIntervalLengthType" final="#all">
  <simpleContent>
    <extension base="decimal">
      <attribute name="unit" type="gml:TimeUnitType" use="required"/>
      <attribute name="radix" type="positiveInteger" use="optional"/>
      <attribute name="factor" type="integer" use="optional"/>
    </extension>
  </simpleContent>
</complexType>
```

```

</extension>
</simpleContent>
</complexType>
```

ISO 11404 syntax specifies the use of a positiveInteger together with appropriate values for radix and factor. The resolution of the time interval is to one radix ^(-factor) of the specified time unit (e.g. unit="second", radix="10", factor="3" specifies a resolution of milliseconds).

The value of the unit is either selected from the units for time intervals from ISO 31-1:1992, or is another suitable unit. The encoding is defined for GML in gml:TimeUnitType:

```

<simpleType name="TimeUnitType">
  <union>
    <simpleType>
      <restriction base="string">
        <enumeration value="year"/>
        <enumeration value="month"/>
        <enumeration value="day"/>
        <enumeration value="hour"/>
        <enumeration value="minute"/>
        <enumeration value="second"/>
      </restriction>
    </simpleType>
    <simpleType>
      <restriction base="string">
        <pattern value="other:\w{2,}"/>
      </restriction>
    </simpleType>
  </union>
</simpleType>
```

The second component of this union type provides a method for indicating time units other than the six standard units given in the enumeration.

For example, to express a period length of 5 days, 14 hours, and 30 minutes any of the following instances are acceptable:

```

<duration>P5DT14H30M</duration>
<timeInterval unit="hour" radix="10" factor="0">134.5</timeInterval>
<timeInterval unit="other:week" radix="10" factor="0"> 0.800595</timeInterval>
```

## 14.3 Temporal topology schema

Temporal topology is described in terms of time complexes, nodes, and edges, and the connectivity between these. The temporal context of features that does not relate directly to position in time is described using connectivity relationships among instants and periods. These may incidentally provide information about the ordering of objects in time. Temporal topology does not directly provide information about temporal position. It is used in the case of describing a lineage or a history (ex. a family tree expressing evolution of species, an ecological cycle, a lineage of lands or buildings, or a history of separation and merger of administrative boundaries). This clause specifies the temporal topology as temporal characteristics of features in compliance with ISO 19108 – Temporal Schema.

### 14.3.1 Overview

The temporal topology types and elements are described in the schema listed in Annex C. The temporal topology schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:temporalTopology:v3.1.0

### 14.3.2 Temporal topology objects

A temporal topology object shall be a temporal element that describes the order of features or feature properties as temporal characteristics of features. The two temporal topology objects are primitive and complex. Annex C defines the schema for temporal topology objects.

As time is one dimensional topological space, temporal topology primitives shall be a time node corresponding to an instant, and a time edge corresponding to a period. A time node is an abstraction of an event that happened at a certain instant as a start or an end of more than one states. A state is a condition — a characteristic of a feature or data set that persists for a period. A “static feature” in this standard means a feature that holds the consistent identifier during its life span. Time edge is an abstraction of a state, and associates with time nodes representing its start and end. However, temporal topology primitives do not directly indicate “when” or “how long.” Time node may not be a start or an end of a time edge in the case of describing the event not associating with states. Such a node is called as an isolated node.

Topological complex is a collection of topological primitives that is closed under the boundary operation. A temporal topological complex shall be a connected acyclic directed graph composed of time edges and time nodes. A minimum temporal topological complex is a time edge with two time nodes at its both ends.

Example A lifecycle of a building can be described as a sequence of stages: plan, designing, construction, utilization, disposal and demolition. Each stage can be represented as a time edge. The boundary of each stage describing as a time node represents an event of decision-making, which terminates the stage and also originates the next stage. Thus, a lifecycle of a building is described as a temporal topology complex composed of a sequence of time edges connected with time nodes.

#### 14.3.2.1 `gml:_TimeTopologyPrimitive`

Temporal topology primitives shall imply the ordering information between features or feature properties. The temporal connection of features can be examined, if they have temporal topology primitives as values of their properties. Usually, an instantaneous feature associates with a time node, and a static feature associates with a time edge. A feature with both modes associates with the temporal topology primitive: a supertype of time nodes and time edges.

The abstract element `TimeTopologyPrimitive` acts as the head of a substitution group for topological temporal primitives. It is defined in the schema as follows:

```
<element name="_TimeTopologyPrimitive" type="gml:AbstractTimeTopologyPrimitiveType" abstract="true"
substitutionGroup="gml:_TimePrimitive"/>
```

`gml:_TimeTopologyPrimitive` may be used in any position that a `gml:_TimePrimitive` is valid. Its content model is defined as follows:

```
<complexType name="AbstractTimeTopologyPrimitiveType" abstract="true">
<complexContent>
<extension base="gml:AbstractTimePrimitiveType">
<sequence>
<element name="complex" type="gml:ReferenceType" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>
```

A topological primitive is always connected to one or more other topological primitives, and is, therefore, always a member of a topological complex. In a GML instance, this will often be indicated by the primitives being described by elements that are descendants of an element describing a complex. However, in order to support the case where a temporal topological primitive is described in another context, the optional “complex” property is provided, which carries a reference to the parent temporal topological complex.

#### 14.3.2.2 gml:TimeTopologyPrimitivePropertyType

gml:TimeTopologyPrimitivePropertyType provides for associating a gml:\_TimeTopologyPrimitive with an object:

```
<complexType name="TimeTopologyPrimitivePropertyType">
  <sequence>
    <element ref="gml:_TimeTopologyPrimitive" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

#### 14.3.2.3 gml:TimeTopologyComplex

Temporal topology complex shall be the connected acyclic directed graph composed of temporal topological primitives, i.e. time nodes and time edges. Because a time edge cannot exist without two time nodes on its boundaries, static features shall have time edges from temporal topology complex as the values of their temporal properties, regardless of explicit declarations.

Temporal topology complex expresses a linear or a non-linear graph. A temporal linear graph, composed of a sequence of time edges, provides a lineage described only by “substitution” of feature instances or feature element values. A time node as the start or the end of the graph connects with at least one time edge. A time node other than the start and the end shall connect to at least two time edges: one of starting from the node, and another ending at the node.

The element gml:TimeTopologyComplex is declared as follows:

```
<element name="TimeTopologyComplex" type="gml:TimeTopologyComplexType"
substitutionGroup="gml:_TimeComplex"/>
```

gml:TimeTopologyComplex may be used in any position that a gml:\_TimeComplex is valid. Its content model is defined as follows:

```
<complexType name="TimeTopologyComplexType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractTimeComplexType">
      <sequence>
        <element name="primitive" type="gml:TimeTopologyPrimitivePropertyType" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A temporal topological complex is a set of connected temporal topological primitives. The member primitives are indicated, either by reference or by value, using the “primitive” property.

#### 14.3.2.4 gml:TimeNode

A time node is a zero-dimensional topological primitive that represents an identifiable node in time (it is equivalent to a point in space). A node may act as the termination or initiation of any number of time edges. A time node may be realised as a geometry, its position, whose value is a time instant.

The element gml:TimeNode is declared as follows:

```
<element name="TimeNode" type="gml:TimeNodeType" substitutionGroup="gml:_TimeTopologyPrimitive"/>
```

gml:TimeNode may be used in any position that a gml:\_TimeTopologyPrimitive is valid. Its content model is defined as follows:

```
<complexType name="TimeNodeType">
```

```

<complexContent>
  <extension base="gml:AbstractTimeTopologicalPrimitiveType">
    <sequence>
      <element name="previousEdge" type="gml:TimeEdgePropertyType" minOccurs="0" maxOccurs="unbounded"/>
      <element name="nextEdge" type="gml:TimeEdgePropertyType" minOccurs="0" maxOccurs="unbounded"/>
      <element name="position" type="gml:TimeInstantPropertyType" minOccurs="0"/>
    </sequence>
  </extension>
</complexContent>
</complexType>

```

#### 14.3.2.5 gml:TimeNodePropertyType

gml:TimeNodePropertyType provides for associating a gml:TimeNode with an object:

```

<complexType name="TimeNodePropertyType">
  <sequence>
    <element ref="gml:TimeNode" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

```

#### 14.3.2.6 gml:TimeEdge

A time edge is a one-dimensional topological primitive. It is an open interval that starts and ends at a node. The edge may be realised as a geometry whose value is a time period

The element gml:TimeEdge is declared as follows:

```
<element name="TimeEdge" type="gml:TimeEdgeType" substitutionGroup="gml:_TimeTopologyPrimitive"/>
```

gml:TimeEdge may be used in any position that a gml:\_TimeTopologyPrimitive is valid. Its content model is defined as follows:

```

<complexType name="TimeEdgeType">
  <complexContent>
    <extension base="gml:AbstractTimeTopologyPrimitiveType">
      <sequence>
        <element name="start" type="gml:TimeNodePropertyType"/>
        <element name="end" type="gml:TimeNodePropertyType"/>
        <element name="extent" type="gml:TimePeriodPropertyType" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

#### 14.3.2.7 gml:TimeEdgePropertyType

gml:TimeEdgePropertyType provides for associating a gml:TimeEdge with an object:

```

<complexType name="TimeEdgePropertyType">
  <sequence>
    <element ref="gml:TimeEdge" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

```

#### 14.3.2.8 gml:SuccessionType

A temporal non-linear graph is a network composed of time edges. This provides a framework for describing successions of feature instances or feature property values described by “substitution”, “division” and “fusion”.

In support of application schemas that require this capability, the gml:SuccessionType is provided to ensure a consistent terminology. It is defined as follows:

```
<simpleType name="SuccessionType">
  <restriction base="string">
    <enumeration value="substitution"/>
    <enumeration value="division"/>
    <enumeration value="fusion"/>
    <enumeration value="initiation"/>
  </restriction>
</simpleType>
```

Note that successions may occur not only in the geographic data. For example, a restructuring of organization and an explication of philosophy are not always related to positions on the Earth, but frequently be accompanied with “division” or “fusion.”

## 14.4 Temporal reference systems

### 14.4.1 Overview

A value in the time domain is measured relative to a temporal reference system. Common types of reference systems include calendars, ordinal temporal reference systems, and temporal coordinate systems (time elapsed since some epoch, e.g. UNIX time, GPS time). The primary temporal reference system for use with geographic information is the Gregorian Calendar and 24 hour local or Coordinated Universal Time (UTC), but special applications may entail the use of alternative reference systems. The Julian day numbering system is a temporal coordinate system that has an origin earlier than any known calendar, at noon on 1 January 4713 BC in the Julian proleptic calendar, and is useful in transformations between dates in different calendars.

In GML seven concrete elements are used to describe temporal reference systems: TimeReferenceSystem, TimeCoordinateSystem, TimeCalendar, TimeCalendarEra, TimeClock, TimeOrdinalReferenceSystem, and TimeOrdinalEra.

The types and elements for temporal reference systems are described in the schema listed in Annex C. The schema is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:temporalReferenceSystems:v3.1.0

### 14.4.2 Basic temporal reference system

#### 14.4.2.1 gml:TimeReferenceSystem

A reference system is characterized in terms of its domain of validity: the spatial and temporal extent over which it is applicable. The basic GML element for temporal reference systems is gml:TimeReferenceSystem. Its content model simply extends gml:DefinitionType (see clause 15.2.1) with one property: domainOfValidity. In the schema this is implemented as follows:

```
<element name="TimeReferenceSystem" type="gml:TimeReferenceSystemType" substitutionGroup="gml:_GML"/>
```

gml:TimeReferenceSystem may be used in any position that a GML Object (gml:\_GML) is valid. Its content model is defined as follows:

```
<complexType name="TimeReferenceSystemType">
  <complexContent>
    <extension base="gml:DefinitionType">
      <sequence>
        <element name="domainOfValidity" type="string"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```
</complexType>
```

The standard properties of a GML object are inherited from `gml:AbstractGMLType`. This element might appear in an instance document as follows:

```
<gml:TimeReferenceSystem gml:id="JulianCalendar">
  <gml:description xlink:href="http://aa.usno.navy.mil/data/docs/JulianDate.html"/>
  <gml:name>Julian Calendar</gml:name>
  <gml:domainOfValidity>Western Europe</gml:domainOfValidity>
</gml:TimeReferenceSystem>
```

### 14.4.3 Time Coordinate Systems

A temporal coordinate system shall be based on a continuous interval scale defined in terms of a single time interval.

#### 14.4.3.1 gml:TimeCoordinateSystem

The element `gml:TimeCoordinateSystem` is declared as follows:

```
<element name="TimeCoordinateSystem" type="gml:TimeCoordinateSystemType"
  substitutionGroup="gml:TimeReferenceSystem"/>
```

`gml:TimeCoordinateSystem` may be used in any position that a `gml:TimeReferenceSystem` is valid. Its content model is defined as follows:

```
<complexType name="TimeCoordinateSystemType">
  <complexContent>
    <extension base="gml:TimeReferenceSystemType">
      <sequence>
        <choice>
          <element name="originPosition" type="gml:TimePositionType"/>
          <element name="origin" type="gml:TimeInstantPropertyType"/>
        </choice>
        <element name="interval" type="gml:TimeIntervalLengthType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

The model follows TM\_CoordinateSystem from ISO 19108, except in the following details:

- the origin is specified either using the property `gml:originPosition` whose value is a direct time position (clause 14.2.3.6), or using the property `gml:origin` whose model is `gml:TimeInstantPropertyType` (clause 14.2.3.3); this permits more flexibility in representation and also supports referring to a value fixed elsewhere
- the interval uses `gml:TimeIntervalLengthType`, defined in clause 14.2.3.7.

Coordinate systems might be described in data instances as follows:

```
<gml:TimeCoordinateSystem gml:id="Laser36">
  <gml:description>Time scale used during a laser experiment</gml:description>
  <gml:name>Laser timescale 36</gml:name>
  <gml:domainOfValidity>Laser laboratory</gml:domainOfValidity>
  <gml:origin>
    <gml:TimeInstant>
      <gml:timePosition>2002-11-28T12:50:00+08:00</gml:timePosition>
    </gml:TimeInstant>
  </gml:origin>
  <gml:interval unit="second" radix="10" factor="12">1.0</gml:interval>
</gml:TimeCoordinateSystem>
```

```

<gml:TimeCoordinateSystem gml:id="geologyMa">
  <gml:name>Geological time system</gml:name>
  <gml:domainOfValidity>Earth</gml:domainOfValidity>
  <gml:origin>
    <gml:TimeInstant>
      <gml:description xlink:href="http://www.c14dating.com/agecalc.html">Conventional origin used for carbon dating.  
Equivalent to "present" for other radiometric dating techniques which have much lower precision.</gml:description>
      <gml:timePosition>1950</gml:timePosition>
    </gml:TimeInstant>
    <gml:origin>
      <gml:interval unit="year" radix="10" factor="-6">1.0</gml:interval>
    </gml:origin>
  </gml:TimeCoordinateSystem>

```

#### 14.4.4 Calendars and clocks

Calendars and clocks are both based on interval scales. A calendar is a discrete temporal reference system that provides a basis for defining temporal position to a resolution of one day. A clock provides a basis for defining temporal position within a day. A clock must be used with a calendar in order to provide a complete description of a temporal position within a specific day.

Calendars have a variety of complex internal structures. This schema defines a simple external calendar interface. Every calendar provides a set of rules for composing a calendar date from a set of elements such as year, month, and day. In every calendar, years are numbered relative to the date of a reference event that defines a calendar era. A single calendar may reference more than one calendar era

##### 14.4.4.1 gml:TimeCalendar, gml:TimeCalendarEra

A calendar is a discrete temporal reference system that provides a basis for defining temporal position to a resolution of one day. The element gml:TimeCalendar is declared as follows:

```
<element name="TimeCalendar" type="gml:TimeCalendarType" substitutionGroup="gml:TimeReferenceSystem"/>
```

gml:TimeCalendarEra may be used in any position that a gml:TimeReferenceSystem is valid. Its content model is defined as follows:

```

<complexType name="TimeCalendarType">
  <complexContent>
    <extension base="gml:TimeReferenceSystemType">
      <sequence>
        <element name="referenceFrame" type="gml:TimeCalendarEraPropertyType" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

gml:TimeCalendar adds one property to those inherited from gml:TimeReferenceSystem:

referenceFrame: link to a Calendar Era that it uses;

A gml:TimeCalendar may reference more than one Calendar Era. The referenceFrame element follows the standard GML property model, allowing the association to be instantiated either using an inline description using the gml:TimeCalendarEra element, or a link to a description of the Calendar Era which is explicit elsewhere. The gml:TimeCalendarEra element is declared as follows

```
<element name="TimeCalendarEra" type="gml:TimeCalendarEraType" />
```

Its content model is defined as follows:

```

<complexType name="TimeCalendarEraType">
  <complexContent>
    <extension base="gml:DefinitionType">

```

```

<sequence>
  <element name="referenceEvent" type="gml:StringOrRefType"/>
  <element name="referenceDate" type="gml:CalDate"/>
  <element name="julianReference" type="decimal"/>
  <element name="epochOfUse" type="gml:TimePeriodPropertyType"/>
</sequence>
</extension>
</complexContent>
</complexType>

```

gml:TimeCalendarEra inherits basic properties from gml:DefinitionType (clause 15.2.1), has the following properties:

referenceEvent: Name or description of a mythical or historic event which fixes the position of the base scale of the calendar era. This is given as text or using a link to description held elsewhere

referenceDate: Date of the referenceEvent expressed as a date in the given calendar. In most calendars, this date is the origin (i.e., the first day) of the scale, but this is not always true

julianReference: Julian date that corresponds to the reference date. The Julian day number is an integer value; the Julian date is a decimal value that allows greater resolution. Transforming calendar dates to and from Julian dates provides a relatively simple basis for transforming dates from one calendar to another

epochOfUse: Period for which the calendar era was used as a basis for dating.

#### 14.4.4.2 gml:TimeClock

A clock provides a basis for defining temporal position within a day. A clock must be used with a calendar in order to provide a complete description of a temporal position within a specific day. The element gml:TimeClock is declared as follows:

```
<element name="TimeClock" type="gml:TimeClockType" substitutionGroup="gml:TimeReferenceSystem"/>
```

gml:TimeClock may be used in any position that a gml:TimeReferenceSystem is valid. Its content model is defined as follows:

```

<complexType name="TimeClockType" final="#all">
  <complexContent>
    <extension base="gml:TimeReferenceSystemType">
      <sequence>
        <element name="referenceEvent" type="gml:StringOrRefType"/>
        <element name="referenceTime" type="time"/>
        <element name="utcReference" type="time"/>
        <element name="dateBasis" type="gml:TimeCalendarPropertyType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

gml:TimeClock adds the following properties to those inherited from generic gml:TimeReferenceSystemType:

referenceEvent: Name or description of an event, such as solar noon or sunrise, which fixes the position of the base scale of the clock

referenceTime: time of day associated with the reference event expressed as a time of day in the given clock. The reference time is usually the origin of the clock scale.

UtcReference: 24 hour local or UTC time that corresponds to the reference time.

DateBasis: pointer to the calendars that use this clock.

#### 14.4.5 Time Ordinal Reference Systems

In some applications of geographic information — such as geology and archaeology — relative position in time is known more precisely than absolute time or duration. The order of events in time can be well established, but the magnitude of the intervals between them cannot be accurately determined; in such cases, the use of an ordinal temporal reference system is appropriate. An ordinal temporal reference system is composed of a sequence of named coterminous eras, which may in turn be composed of sequences of member eras at a finer scale, giving the whole a hierarchical structure of eras of varying resolution.

An ordinal temporal reference system whose component eras are not further subdivided is effectively a temporal topological complex constrained to be a linear graph. An ordinal temporal reference system some or all of whose component eras are subdivided is effectively a temporal topological complex with the constraint that parallel branches may only be constructed in pairs where one is a single temporal ordinal era and the other is a sequence of temporal ordinal eras that are called "members" of the "group". This constraint means that within a single temporal ordinal reference system, the relative position of all temporal ordinal eras is unambiguous.

The positions of the beginning and end of a given era may calibrate the relative time scale.

##### 14.4.5.1 **gml:TimeOrdinalReferenceSystem**, **gml:TimeOrdinalEra**

The element **gml:TimeOrdinalReferenceSystem** implements this by adding a set of **gml:component** properties to the generic temporal reference system model. It is declared as follows:

```
<element name="TimeOrdinalReferenceSystem" type="gml:TimeOrdinalReferenceSystemType"
substitutionGroup="gml:TimeReferenceSystem"/>
```

**gml:TimeOrdinalReferenceSystem** may be used in any position that a **gml:TimeReferenceSystem** is valid. Its content model is defined as follows:

```
<complexType name="TimeOrdinalReferenceSystemType">
<complexContent>
<extension base="gml:TimeReferenceSystemType">
<sequence>
<element name="component" type="gml:TimeOrdinalEraPropertyType" maxOccurs="unbounded"/>
</sequence>
</extension>
</complexContent>
</complexType>
```

The content model for **gml:TimeOrdinalEra** follows the pattern of **gml:TimeEdge** (clause 14.3.2.6), inheriting standard properties from **gml:DefinitionType** (clause 15.2.1), and adding **gml:start**, **gml:end** and **gml:extent** properties, a set of **gml:member** properties which indicate ordered **gml:TimeOrdinalEra** elements, and a **gml:group** property which points to the parent era. This is implemented as follows:

```
<element name="TimeOrdinalEra" type="gml:TimeOrdinalEraType"/>

<complexType name="TimeOrdinalEraType">
<complexContent>
<extension base="gml:DefinitionType">
<sequence>
<element name="relatedTime" type="gml:RelatedTimeType" minOccurs="0" maxOccurs="unbounded"/>
<element name="start" type="gml:TimeNodePropertyType"/>
<element name="end" type="gml:TimeNodePropertyType"/>
<element name="extent" type="gml:TimePeriodPropertyType" minOccurs="0"/>
<element name="member" type="gml:TimeOrdinalEraPropertyType" minOccurs="0" maxOccurs="unbounded"/>
<element name="group" type="gml:ReferenceType" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>
```

The model for the component and member properties follows the normal GML property pattern:

```
<complexType name="TimeOrdinalEraPropertyType">
  <sequence minOccurs="0">
    <element ref="gml:TimeOrdinalEra"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

The recursive inclusion of gml:TimeOrdinalEra elements allow the construction of an arbitrary depth hierarchical ordinal reference schema, such that an ordinal era at a given level of the hierarchy includes a sequence of shorter, coterminous ordinal eras.

The example below shows a portion of the geological time scale depicted as an ordinal reference system:

```
<gml:TimeOrdinalReferenceSystem gml:id="GeologicalTimeScale">
  <gml:description xlink:href="ftp://ftp.iugs.org/pub/iugs/iugs_intstratchart.pdf"/>
  <gml:name>Geological time scale</gml:name>
  <gml:domainOfValidity>Earth</gml:domainOfValidity>
  <!-- Earlier eras omitted -->
  <gml:component>
    <gml:TimeOrdinalEra gml:id="Cenozoic">
      <gml:name>Cenozoic Era</gml:name>
      <gml:start xlink:href="#basePaleocene"/>
      <gml:end xlink:href="#now"/>
      <gml:member>
        <gml:TimeOrdinalEra gml:id="Tertiary">
          <gml:name>Tertiary Period</gml:name>
          <gml:start xlink:href="#baseTertiary"/>
          <gml:end xlink:href="#basePleistocene"/>
          <gml:member>
            <gml:TimeOrdinalEra gml:id="Paleogene">
              <gml:name>Paleogene sub-period</gml:name>
              <gml:start>
                <gml:TimeInstant gml:id="basePaleogene">
                  <gml:timePosition frame="#geologyMa">65.0</gml:timePosition>
                </gml:TimeInstant>
              </gml:start>
              <gml:end xlink:href="#baseNeogene"/>
              <gml:member>
                <gml:TimeOrdinalEra gml:id="Paleocene">
                  <gml:name>Paleocene Epoch</gml:name>
                  <gml:start xlink:href="#basePaleogene"/>
                  <gml:end xlink:href="#baseEocene"/>
                </gml:TimeOrdinalEra>
              </gml:member>
              <gml:member>
                <gml:TimeOrdinalEra gml:id="Eocene">
                  <gml:name>Paleocene Epoch</gml:name>
                  <gml:start>
                    <gml:TimeInstant gml:id="baseEocene">
                      <gml:timePosition frame="#geologyMa">57.8</gml:timePosition>
                    </gml:TimeInstant>
                  </gml:start>
                  <gml:end xlink:href="#baseOligocene"/>
                </gml:TimeOrdinalEra>
              </gml:member>
              <gml:member>
                <gml:TimeOrdinalEra gml:id="Oligocene">
                  <gml:name>Oligocene Epoch</gml:name>
                  <gml:start>
                    <gml:TimeInstant gml:id="baseOligocene">
                      <gml:timePosition frame="#geologyMa">33.7</gml:timePosition>
                    </gml:TimeInstant>
```

```

</gml:start>
<gml:end xlink:href="#baseNeogene"/>
</gml:TimeOrdinalEra>
<gml:member>
</gml:TimeOrdinalEra>
<gml:member>
<!-- Neogene sub-period and Quaternary period omitted -->
</gml:TimeOrdinalEra>
<gml:member>
</gml:TimeOrdinalEra>
<gml:component>
</gml:TimeOrdinalReferenceSystem>

```

Note that the use of references on various begin and end elements allows the position of the boundaries between eras to be recorded once and then re-used many times as appropriate, corresponding to a non-linear graph when appropriate. All positions refer to a frame “geologyMa” which would be defined as a temporal coordinate system (e.g see clause 14.4.3).

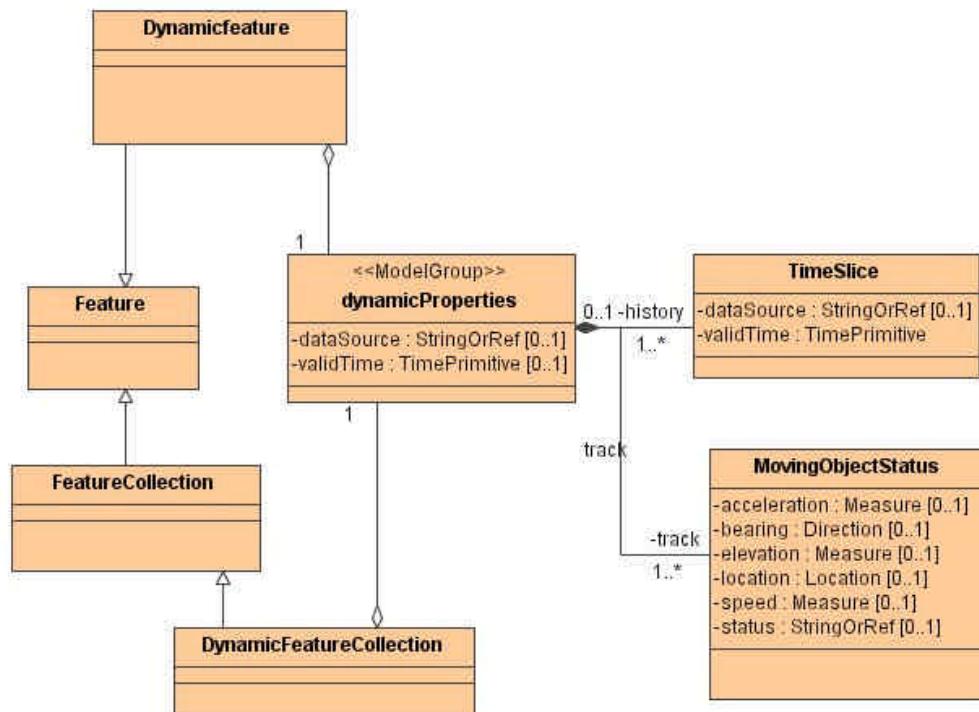
## 14.5 Representing dynamic features

### 14.5.1 Overview

A number of types and relationships are defined to represent the time-varying properties of geographic features. The dynamic feature schema is listed in Annex C; it is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:dynamicFeature:v3.1.0

In a comprehensive treatment of spatiotemporal modeling, Langran [see bibliography] distinguished three principal temporal entities: *states*, *events*, and *evidence*; this schema incorporates elements for each. A simple UML model for dynamic features is shown in Figure.



### Figure 3 – Dynamic Features

#### 14.5.2 **gml:dataSource**

In GML, **evidence** is represented by a simple **dataSource** property that indicates the source of the temporal data (e.g. human observer, *in situ* sensor).

```
<element name="dataSource" type="gml:StringOrRefType" abstract="true"/>
```

#### 14.5.3 Dynamic Properties

A convenience group **gml:dynamicProperties** is defined in the schema as follows:

```
<group name="dynamicProperties">
<sequence>
<element ref="gml:validTime" minOccurs="0"/>
<element ref="gml:history" minOccurs="0"/>
<element ref="gml:dataSource" minOccurs="0"/>
</sequence>
</group>
```

This allows an application schema developer to include dynamic properties in a content model in a standard fashion. The **gml:validTime** property is declared in **temporal.xsd** (described in clause 14.2.2.3 above). The other properties are declared elsewhere in this clause.

#### 14.5.4 Dynamic Features

**States** are captured by time-stamped instances of a feature or feature collection. The content model for a dynamic feature extends the standard **AbstractFeatureType** and **FeatureCollectionType** with the **gml:dynamicProperties** model group defined above:

```
<complexType name="DynamicFeatureType">
<complexContent>
<extension base="gml:AbstractFeatureType">
<group ref="gml:dynamicProperties"/>
</extension>
</complexContent>
</complexType>

<complexType name="DynamicFeatureCollectionType">
<complexContent>
<extension base="gml:FeatureCollectionType">
<group ref="gml:dynamicProperties"/>
</extension>
</complexContent>
</complexType>
```

Each time-stamped instance represents a ‘snapshot’ of a feature. The dynamic feature classes will normally be extended to suit particular applications. A dynamic feature bears either a time stamp or a history.

**NOTE** A history consists of a set of **gml:TimeSlices** and such time slices can contain any time varying properties. We might for example use such a mechanism to describe a feature with one property that varies in time.

#### 14.5.5 **gml:DynamicFeatureCollection**

A **DynamicFeatureCollection** is a **FeatureCollection** that has a **validTime** property (i.e. is a snapshot of the **FeatureCollection**) or which has a **gml:history** property that contains one or more **gml:TimeSlices** each of which contain values of the time varying properties of the **FeatureCollection**. Note that the **DynamicFeatureCollection** can be one of the following:

1. A FeatureCollection which consists of static feature members (members do not change in time) but which has properties of the collection object as a whole that do change in time (e.g. described by a history). An example might be a Train. The Train is a `gml:FeatureCollection`. The position and speed of the train are time varying and could be captured in the history of the Train. The `featureMembers` of the Train are the individual cars including the locomotive. The properties of the cars are static such as the position of the car in the train (we ignore any re-organization of the train in this example), the cargo, the make of the car and its type (e.g. grain car, oil car etc.).
2. A FeatureCollection which consists of dynamic feature members (the members are `DynamicFeatures`) but which also has properties of the collection as a whole that vary in time. An example could be a collection of sail boats in a yachting race. The sail boats may disappear from the race or reappear. The area encompassing the boats in the race (think of a race like the Vendée Globe) would be time variant.

**NOTE** One can also have a FeatureCollection with dynamic feature members but such that the properties of the collection as a whole are static. This might also be applied to the sail boat race where we only have properties like the organization committee, and the location of the starting point and finish line.

#### 14.5.6 `gml:_TimeSlice`

To describe an *event* — an action that occurs at an instant or over an interval of time — GML provides the abstract **TimeSlice** element, which is declared in the schema as follows:

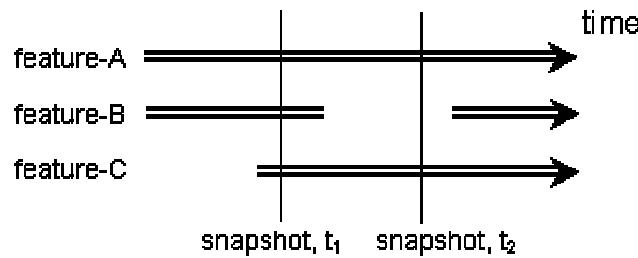
```
<element name="_TimeSlice" type="gml:AbstractTimeSliceType" abstract="true" substitutionGroup="gml:_GML"/>

<complexType name="AbstractTimeSliceType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <sequence>
        <element ref="gml:validTime"/>
        <element ref="gml:dataSource" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A timeslice encapsulates the time-varying properties of a dynamic feature -- it must be extended to represent a time stamped projection of a specific feature. The `dataSource` property describes how the temporal data was acquired.

A TimeSlice instance is a GML object that encapsulates updates of the dynamic—or volatile—properties that reflect some change event; it thus includes only those feature properties that have actually changed due to some process. For example, suppose that ownership of a building changes and it is renamed. If no other building properties have changed, then the event will only include the updated name. The `TimeSlice` class basically provides a facility for *attribute-level* time stamping, in contrast to the *object-level* time stamping of dynamic feature instances.

The time slice can thus be viewed as event or process-oriented, whereas a snapshot is more state or structure-oriented. A timeslice has richer causality, whereas a snapshot merely portrays the status of the whole. For example, a feature collection might have a 'life cycle' represented by a sequence of snapshots (Figure Figure 14.4.6-2).



**Figure 14.4.6-2—The life cycle of a feature collection**

At instant  $t_1$ , feature-A, feature-B, and feature-C are all members of the collection. However, at instant  $t_2$  only feature-A and feature-B are members. Closer examination of the history of feature-B will reveal its ephemeral nature (e.g. a building is dismantled and reconstructed on a seasonal basis).

#### 14.5.7 **gml:MovingObjectStatus**

The **gml:MovingObjectStatus** element is one example of how the **gml:TimeSlice** may be extended. This element provides a standard method to capture a record of the status of a moving object. It is declared as follows:

```
<element name="MovingObjectStatus" type="gml:MovingObjectStatusType" substitutionGroup="gml:TimeSlice"/>

<complexType name="MovingObjectStatusType">
  <complexContent>
    <extension base="gml:AbstractTimeSliceType">
      <sequence>
        <element ref="gml:position"/>
        <element name="speed" type="gml:MeasureType" minOccurs="0"/>
        <element name="bearing" type="gml:DirectionPropertyType" minOccurs="0"/>
        <element name="acceleration" type="gml:MeasureType" minOccurs="0"/>
        <element name="elevation" type="gml:MeasureType" minOccurs="0"/>
        <element ref="gml:status" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A **gml:MovingObjectStatus** element allows the user to describe the present location, along with the speed, bearing, acceleration and elevation of an object in a particular time slice.

Additional information about the current status of the object can be recorded in the **gml:status** element, declared as follows:

```
<element name="status" type="gml:StringOrRefType"/>
```

#### 14.5.8 **gml:history**

A generic sequence of events constitute a **history** of an object. This GML property element is declared in the schema as follows:

```
<element name="history" type="gml:History.PropertyType"/>

<complexType name="History.PropertyType">
  <sequence>
    <element ref="gml:_TimeSlice" maxOccurs="unbounded"/>
  </sequence>
</complexType>
```

The `gml:history` element contains a set of elements in the substitution group headed by the abstract element `gml:_TimeSlice`, representing the time-varying properties of interest. The `history` property of a dynamic feature associates a feature instance with a sequence of time slices (i.e. change events) that encapsulate the evolution of the feature.

#### 14.5.9 `gml:track`

A track is a specific kind of history. The `gml:track` element is declared in the schema as follows:

```
<element name="track" type="gml:TrackType" substitutionGroup="gml:history"/>

<complexType name="TrackType">
  <complexContent>
    <restriction base="gml:HistoryPropertyType">
      <sequence>
        <element ref="gml:MovingObjectStatus" maxOccurs="unbounded"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
```

If the feature represents a moving object such as a ground vehicle or a ship, then the `track` property comprises a sequence of `MovingObjectStatus` elements. For example, a dynamic feature such as a cyclone may have a `gml:track` property such as shown in the following fragment:

```
<gml:track>
  <gml:MovingObjectStatus>
    <gml:validTime><gml:TimeInstant>
      <gml:timePosition>2005-11-28T13:00:00</gml:timePosition>
    </gml:TimeInstant></gml:validTime>
    <gml:location><gml:Point>
      <gml:pos>140. -35.</gml:pos>
    </gml:Point></gml:location>
    <gml:speed uom="#kph">12.</gml:speed>
    <gml:bearing>
      <gml:CompassPoint>SE</gml:CompassPoint>
    </gml:bearing>
  </gml:MovingObjectStatus>
  <gml:MovingObjectStatus>
    <gml:validTime><gml:TimeInstant>
      <gml:timePosition>2005-11-28T14:00:00</gml:timePosition>
    </gml:TimeInstant></gml:validTime>
    <gml:location><gml:Point>
      <gml:pos>140.1 -34.9</gml:pos>
    </gml:Point></gml:location>
    <gml:speed uom="#kph">23.</gml:speed>
    <gml:bearing>
      <gml:CompassPoint>ESE</gml:CompassPoint>
    </gml:bearing>
  </gml:MovingObjectStatus>
</gml:track>
```

## 15 GML schemas – definitions and dictionaries

### 15.1 Overview

Many applications require definitions of terms, which are used within instance documents as the values of certain properties or as reference information to tie properties to standard information values in some way. Units of measure, and descriptions of measurable phenomena, are two particular examples.

It will often be convenient to use definitions provided by external authorities. These may already be packaged for delivery in various ways, both online and offline. In order that they may be referred to from GML documents it is generally necessary that a URI be available for each definition. Where this is the case then it is usually preferable to refer to these directly.

Alternatively, it may be convenient or necessary to capture definitions in XML, either embedded within an instance document containing Features or as a separate document. The definitions may be transcriptions from an external source, or may be new definitions for a local purpose. In order to support this case, some simple components are provided in GML in the form of

- a) a generic Definition, which may serve as the basis for more specialized definitions
- b) a generic Dictionary, also known as DefinitionCollection, which allows a set of definitions or references to definitions to be collected

These components may be used directly, but also serve as the basis for more specialised definition elements in GML, in particular: coordinate operations (clause 12), coordinate reference systems (clause 12), datums (clause 12), temporal reference systems (clause 14), units of measure (clause 16).

Note that the GML definition and dictionary components implement a simple nested hierarchy of definitions with identifiers. The latter provide handles which may be used in the description of more complex relationships between terms. However, the GML dictionary components are not intended to provide direct support for complex taxonomies, ontologies or thesauri. Specialised XML tools are available to satisfy the more sophisticated requirements.

## 15.2 Dictionary schema

The dictionary schema is listed in Annex C; it is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:dictionary:v3.1.0

### 15.2.1 gml:Definition, gml:DefinitionType

The basic Definition element is declared in the schema as follows:

```
<element name="Definition" type="gml:DefinitionType" substitutionGroup="gml:_GML"/>

<complexType name="DefinitionType">
  <complexContent>
    <restriction base="gml:AbstractGMLType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:description" minOccurs="0"/>
        <element ref="gml:name" minOccurs="1" maxOccurs="unbounded"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
```

The content model for a generic definition is a simple derivation from AbstractGMLType. Since it is necessary to be able to refer to a definition, the gml:id attribute which provides the database handle for a GML object is mandatory. The "description" element shall hold the definition if this can be captured in a simple text string, or may carry a link to a description elsewhere. The "name" elements shall provide one or more terms and synonyms for which this is the definition. The "metaDataProperty" elements may be used to reference or include more information about this definition.

### **15.2.2 `gml:Dictionary`, `gml:DefinitionCollection`, `gml:DictionaryType`**

Sets of definitions may be collected into dictionaries or collections. These are declared in the schema as follows:

```
<element name="Dictionary" type="gml:DictionaryType" substitutionGroup="gml:Definition"/>  
<element name="DefinitionCollection" type="gml:DictionaryType" substitutionGroup="gml:Definition"/>  
  
<complexType name="DictionaryType">  
  <complexContent>  
    <extension base="gml:DefinitionType">  
      <choice minOccurs="0" maxOccurs="unbounded">  
        <element ref="gml:dictionaryEntry"/>  
        <element ref="gml:indirectEntry"/>  
      </choice>  
    </extension>  
  </complexContent>  
</complexType>
```

A Dictionary is a non-abstract collection of definitions. These definitions are referenced from other places, in the same and different XML documents.

The Dictionary content model adds a list of dictionaryEntry and indirectEntry properties that contain or reference Definition objects. A database handle (gml:id attribute) is required, in order that this collection may be referred to. The standard metaDataProperty, description and name are available to reference or contain more information about this dictionary. The inherited "description" element can be used for a description of this dictionary. The inherited "name" element can be used for the name(s) of this dictionary.

### **15.2.3 `gml:dictionaryEntry`, `gml:definitionMember`, `gml:DictionaryEntryType`**

These elements contain or refer to the definitions which are members of this dictionary. The element `gml:dictionaryEntry` is declared as follows:

```
<element name="dictionaryEntry" type="gml:DictionaryEntryType"/>  
  
<element name="definitionMember" type="gml:DictionaryEntryType" substitutionGroup="gml:dictionaryEntry"/>  
  
<complexType name="DictionaryEntryType">  
  <sequence>  
    <element ref="gml:Definition" minOccurs="0"/>  
  </sequence>  
  <attributeGroup ref="gml:AssociationAttributeGroup"/>  
</complexType>
```

The content model follows the standard GML property pattern, so a `gml:dictionaryEntry` may either contain or refer to a single `gml:Definition`. Since `gml:Dictionary` is substitutable for `gml:Definition`, the content of an entry can itself be a lower level dictionary or definition collection.

Note that if the value is provided by reference, this definition does *not* carry a handle (`gml:id`) in this context, so does *not* allow external references to this specific definition in this context. When used in this way the referenced definition will usually be in a dictionary in the same XML document.

### **15.2.4 `gml:indirectEntry`, `gml:IndirectEntryType`, `gml:DefinitionProxy`, `gml:DefinitionProxyType`**

If a definition shall be included by reference, in its context within the current collection, then `gml:indirectEntry` shall be used. This is declared as follows:

```
<element name="indirectEntry" type="gml:IndirectEntryType"/>  
  
<complexType name="IndirectEntryType">
```

```
<sequence>
  <element ref="gml:DefinitionProxy"/>
</sequence>
</complexType>
```

A gml:indirectEntry contains a proxy object gml:DefinitionProxy which is declared as follows:

```
<element name="DefinitionProxy" type="gml:DefinitionProxyType" substitutionGroup="gml:Definition"/>

<complexType name="DefinitionProxyType">
  <complexContent>
    <extension base="gml:DefinitionType">
      <sequence>
        <element ref="gml:definitionRef"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

A gml:DefinitionProxy carries a mandatory handle (gml:id), and contains a reference to a definition represented elsewhere. This entry is expected to be convenient in allowing multiple elements in one XML document to contain short (abbreviated XPointer) references, which are resolved to an external definition provided in a Dictionary element in the same XML document.

The reference is carried by a gml:definitionRef element which is declared as follows.

```
<element name="definitionRef" type="gml:ReferenceType"/>
```

This uses the gml:ReferenceType which is described in clause 7.5.3.2. The remote entry referenced can be in a dictionary in the same or different XML document.

The following examples shows two instances of dictionaries:

```
<gml:Dictionary gml:id="rockTypes">
  <gml:description>A simple dictionary of rock types using components from gmlBase</gml:description>
  <gml:name>Example dictionary of rock types</gml:name>
  <gml:dictionaryEntry>
    <gml:Definition gml:id="granite">
      <gml:description>A igneous rock normally composed of quartz, two feldspars and optional mica</gml:description>
      <gml:name>Granite</gml:name>
    </gml:Definition>
  </gml:dictionaryEntry>
  <gml:dictionaryEntry>
    <gml:Definition gml:id="sst">
      <gml:description>A detrital sedimentary rock normally composed of siliceous grains</gml:description>
      <gml:name>Sandstone</gml:name>
    </gml:Definition>
  </gml:dictionaryEntry>
  <gml:indirectEntry>
    <gml:DefinitionProxy gml:id="l1st">
      <gml:definitionRef xlink:href="http://my.big.org/definitions/geology/limestone"/>
    </gml:DefinitionProxy>
  </gml:indirectEntry>
</gml:Dictionary>

<gml:Dictionary gml:id="AbridgedGMLdictionary">
  <gml:description>Abridged GML dictionary. See section 4 Terms and definitions for the rest.</gml:description>
  <gml:name>GML dictionary</gml:name>
  <gml:dictionaryEntry>
    <gml:Definition gml:id="association">
      <gml:description>A structural relationship that describes a set of links, in which a link is a connection among objects; the semantic relationship between two or more classifiers that involves the connections among their instances (Booch, 1999).</gml:description>
      <gml:name>association</gml:name>
    </gml:Definition>
```

```

</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DefinitionCollection gml:id="attribute">
    <gml:description>named property</gml:description>
    <gml:name>attribute</gml:name>
    <gml:definitionMember>
      <gml:Definition gml:id="attribute_in_UML">
        <gml:description>Named property of a class that describes the range of values that instances of the property may hold. (Booch,1999)</gml:description>
        <gml:name codeSpace="UML">attribute</gml:name>
      </gml:Definition>
    </gml:definitionMember>
    <gml:definitionMember>
      <gml:Definition gml:id="attribute_in_XML">
        <gml:description>An information item in the XML Information Set [InfoSet] </gml:description>
        <gml:name codeSpace="XML">attribute</gml:name>
      </gml:Definition>
    </gml:definitionMember>
    <gml:definitionMember>
      <gml:Definition gml:id="property_in_GML">
        <gml:description>A characteristic of a GML object.</gml:description>
        <gml:name codeSpace="GML">property</gml:name>
        <gml:definitionMember>
          <gml:Definition gml:id="property_in_UML">
            <gml:description>A characteristic of a UML object.</gml:description>
            <gml:name codeSpace="UML">property</gml:name>
            <gml:dictionaryEntry xlink:href="#association"/>
            <gml:dictionaryEntry xlink:href="#attribute_in_UML"/>
          </gml:Definition>
        </gml:definitionMember>
      </gml:Definition>
    </gml:definitionMember>
    <gml:definitionMember>
      <gml:DefinitionProxy gml:id="property_proxy">
        <gml:name>property</gml:name>
        <gml:definitionRef xlink:href="#property"/>
      </gml:DefinitionProxy>
    </gml:definitionMember>
  </gml:DefinitionCollection>
</gml:dictionaryEntry>
<gml:indirectEntry>
  <gml:DefinitionProxy gml:id="property_proxy">
    <gml:name>property</gml:name>
    <gml:definitionRef xlink:href="#property"/>
  </gml:DefinitionProxy>
</gml:indirectEntry>
</gml:Dictionary>

```

### 15.2.5 Using Definitions and Dictionaries

Dictionaries and Definitions are GML Objects, so may be found in independent GML data instance documents.

In application schemas it might be useful to attach a Dictionary or Definition Collection, or Definitions to a feature collection in order to record definitions used in properties of members of the collection.

## 16 GML schemas – units, measures and values

### 16.1 Introduction

Several GML schemas introduce components that concern or require quantitative values which use a reference scale or units of measure. In Clause 7.2, basicTypes.xsd provides MeasureType, MeasureListType

and MeasureOrNullListType definitions to enable GML properties and objects to carry units of measure, according to the following pattern:

```
abc:length uom = "#m">>100</abc:length>
```

The attribute “uom” means “unit of measure” and holds a URI which refers to a resource that defines the units. The following schema documents describe components concerning this topic:

- units.xsd declares a set of components for defining units of measure
- measures.xsd declares a set of typed measures;
- valueObjects.xsd describes structures for aggregates and lists of measures.

The schema documents units.xsd, measures.xsd, and valueObjects.xsd define components for the description of measurement scales, typed measures, and generic (possibly complex) values, respectively.

## 16.2 Units schema

Several GML schemas define components that concern or require a reference scale or units of measure. Units are required for quantities that may occur as values of properties of feature types, as the results of observations, in the range parameters of a coverage, and for measures used in Coordinate Reference System definitions. The schema document units.xsd defines components to support the definition of units of measure. The units schema is listed in Annex C; it is identified by the following location-independent name (using URN syntax):

```
urn:opengis:specification:gml:schema-xsd:units:v3.0
```

The basic unit definition is an extension of the general gml:Definition element defined in clause 15.2. Three specialized elements for unit definition are further derived from this.

This model is based on the SI system of units [ISO 1000], which distinguishes between Base Units and Derived Units.

- **Base Units** are the preferred units for a set of orthogonal fundamental quantities which define the particular system of units, which may not be derived by combination of other base units.
- **Derived Units** are the preferred units for other quantities in the system, which may be defined by algebraic combination of the base units.

In some application areas **Conventional units** are used, which may be converted to the preferred units using a scaling factor or a formula which defines a re-scaling and offset. The set of preferred units for all quantity types in a particular system of units is composed of the union of its base units and derived units.

### 16.2.1 Using Unit Definitions

Unit definitions are substitutable for the gml:Definition element declared as part of the dictionary model. A dictionary that contains only unit definitions and references to unit definitions is a units dictionary.

### 16.2.2 gml:unitOfMeasure, gml:UnitOfMeasureType

The element gml:unitOfMeasure is a property element to refer to a unit of measure. It is declared in the schema as follows:

```
<element name="unitOfMeasure" type="gml:UnitOfMeasureType"/>
<complexType name="UnitOfMeasureType">
  <sequence/>
  <attribute name="uom" type="anyURI" use="required"/>
```

```
</complexType>
```

This is an empty element which carries a reference to a unit of measure definition. This element may appear in a data instance as follows:

```
<unitOfMeasure uom="#m"/>  
<unitOfMeasure uom="http://my.standards.org/units/length/metre"/>
```

If the unit of measure definition is within the same XML document, the URI may be a barename Xpointer, in which the "#" symbol is prepended to the value of the attribute of XML type ID carried by the definition (e.g. gml:id). For convenience, the ID may be a mnemonic abbreviation of the unit name. Thus, in the first case the reference is to an element in the same document which carries gml:id="m". In the second example the reference is to a definition provided by an external service.

### 16.2.3 gml:UnitDefinition, gml:UnitDefinitionType

A UnitDefinition is a general definition of a unit of measure. This generic element is used only for units for which no relationship with other units or units systems is known. It is declared in the schema as follows:

```
<element name="UnitDefinition" type="gml:UnitDefinitionType" substitutionGroup="gml:Definition"/>  
  
<complexType name="UnitDefinitionType">  
  <complexContent>  
    <extension base="gml:DefinitionType">  
      <sequence>  
        <element ref="gml:quantityType"/>  
        <element ref="gml:catalogSymbol" minOccurs="0"/>  
      </sequence>  
    </extension>  
  </complexContent>  
</complexType>
```

The content model of gml:UnitDefinition adds two additional properties to gml:Definition (described in clause 15.2.1), gml:quantityType and gml:catalogSymbol. gml:quantityType

### 16.2.4 gml: quantityType

The gml:quantityType property indicates the phenomenon to which the units apply. It is declared as follows:

```
<element name="quantityType" type="gml:StringOrRefType"/>
```

This element contains an informal description of the phenomenon or type of quantity that is measured or observed. For example, "length", "angle", "time", "pressure", or "temperature". When the quantity is the result of an observation or measurement, this term is known as Observable Type or Measurand.

### 16.2.5 gml:catalogSymbol

The catalogSymbol is the preferred lexical symbol used for this unit of measure. It is declared as follows:

```
<element name="catalogSymbol" type="gml:CodeType"/>
```

The "codeSpace" attribute in "CodeType" identifies a namespace for the catalog symbol value, and might reference the external catalog. The "string" value in "CodeType" contains the value of a symbol that should be unique within this catalog namespace. This symbol often appears explicitly in the catalog, but it could be a combination of symbols using a specified algebra of units. For example, the symbol "cm" might indicate that it is the "m" symbol combined with the "c" prefix.

### 16.2.6 gml:BaseUnit, gml:BaseUnitType, gml:unitsSystem

A base unit is a unit of measure that cannot be derived by combination of other base units within a particular system of units. For example, in the SI system of units, the base units are metre, kilogram, second, Ampere, Kelvin, mole, and candela, for the quantity types length, mass, time interval, electric current, thermodynamic temperature, amount of substance and luminous intensity, respectively.

This is supported using the gml:BaseUnit element which is declared as follows:

```
<element name="BaseUnit" type="gml:BaseUnitType" substitutionGroup="gml:UnitDefinition"/>

<complexType name="BaseUnitType">
  <complexContent>
    <extension base="gml:UnitDefinitionType">
      <sequence>
        <element name="unitsSystem" type="gml:ReferenceType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

gml:BaseUnit extends generic gml:UnitDefinition with the property gml:unitsSystem, which carries a reference to the units system to which this base unit is asserted to belong.

### 16.2.7 gml:DerivedUnit, gml:DerivedUnitType

Derived units are defined by combination of other units. Derived units are used for quantities other than those corresponding to the base units, such as hertz ( $s^{-1}$ ) for frequency, Newton ( $kg \cdot m/s^2$ ) for force. Derived units based directly on base units are usually preferred for quantities other than the fundamental quantities within a system. If a derived unit is not the preferred unit, the ConventionalUnit element should be used instead. The gml:DerivedUnit element is declared as follows:

```
<element name="DerivedUnit" type="gml:DerivedUnitType" substitutionGroup="gml:UnitDefinition"/>

<complexType name="DerivedUnitType">
  <complexContent>
    <extension base="gml:UnitDefinitionType">
      <sequence>
        <element ref="gml:derivationUnitTerm" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

The gml:DerivedUnit extends gml:UnitDefinition with the property gml:derivationUnitTerms.

### 16.2.8 gml:derivationUnitTerms, gml:DerivationUnitTermType

A set of gml:derivationUnitTerm elements describes a derived unit of measure. Each element carries an integer exponent. The terms are combined by raising each referenced unit to the power of their exponent and forming the product. The element gml:derivationUnitTerm is declared as follows:

```
<element name="derivationUnitTerm" type="gml:DerivationUnitTermType"/>

<complexType name="DerivationUnitTermType">
  <complexContent>
    <extension base="gml:UnitOfMeasureType">
      <attribute name="exponent" type="integer"/>
    </extension>
  </complexContent>
</complexType>
```

```
</complexType>
```

This unit term references another unit of measure (uom) and provides an integer exponent applied to that unit in defining the compound unit. The exponent can be positive or negative, but not zero.

#### 16.2.9 **gml:ConventionalUnit, gml:ConventionalUnitType**

Conventional units that are neither base units nor defined by direct combination of base units are used in many application domains. For example electronVolt for energy, feet and nautical miles for length. In most cases there is a known, usually linear, conversion to a preferred unit which is either a base unit or derived by direct combination of base units. The gml:ConventionalUnit element is declared as follows:

```
<element name="ConventionalUnit" type="gml:ConventionalUnitType" substitutionGroup="gml:UnitDefinition"/>

<complexType name="ConventionalUnitType">
  <complexContent>
    <extension base="gml:UnitDefinitionType">
      <sequence>
        <choice>
          <element ref="gml:conversionToPreferredUnit"/>
          <element ref="gml:roughConversionToPreferredUnit"/>
        </choice>
        <element ref="gml:DerivationUnitTerm" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

The gml:ConventionalUnit extends gml:UnitDefinition with a property that describes a conversion to a preferred unit for this quantity. When the conversion is exact, the element gml:conversionToPreferredUnit should be used, or when the conversion is not exact the element gml:roughConversionToPreferredUnit is available. Both of these elements have the same content model. The gml:derivationUnitTerm property defined above is included to allow a user to optionally record how this unit may be derived from other ("more primitive") units.

#### 16.2.10 **gml:conversionToPreferredUnit, gml:roughConversionToPreferredUnit, gml:ConversionToPreferredUnitType, gml:FormulaType**

The elements gml:conversionToPreferredUnit and gml:roughConversionToPreferredUnit represent parameters used to convert conventional units to preferred units for this quantity type. A preferred unit is either a Base Unit or a Derived Unit that is selected for all values of one quantity type. These conversions are declared in the schema as follows:

```
<element name="conversionToPreferredUnit" type="gml:ConversionToPreferredUnitType"/>

<element name="roughConversionToPreferredUnit" type="gml:ConversionToPreferredUnitType"/>

<complexType name="ConversionToPreferredUnitType">
  <complexContent>
    <extension base="gml:UnitOfMeasureType">
      <choice>
        <element name="factor" type="double"/>
        <element name="formula" type="gml:FormulaType"/>
      </choice>
    </extension>
  </complexContent>
</complexType>
```

The inherited attribute "uom" references the preferred unit that this conversion applies to. The conversion of a unit to the preferred unit for this quantity type is specified by an arithmetic conversion (scaling and/or offset). The content model extends gml:UnitOfMeasureType, which has a mandatory attribute uom which identifies

the preferred unit for the quantity type that this conversion applies to. The conversion is specified by a choice of

gml:factor, which defines the scale factor

gml:formula, which defines a formula

by which a value using the conventional unit of measure can be converted to obtain the corresponding value using the preferred unit of measure. The model for the formula is given as follows:

```
<complexType name="FormulaType">
<sequence>
<element name="a" type="double" minOccurs="0"/>
<element name="b" type="double"/>
<element name="c" type="double"/>
<element name="d" type="double" minOccurs="0"/>
</sequence>
</complexType>
```

This formula defines the parameters of a simple formula by which a value using the conventional unit of measure can be converted to the corresponding value using the preferred unit of measure. The formula element contains elements a, b, c and d, whose values use the XML Schema type "double". These values are used in the formula  $y = (a + bx) / (c + dx)$ , where x is a value using this unit, and y is the corresponding value using the base unit. The elements a and d are optional, and if values are not provided, those parameters are considered to be zero. If values are not provided for both a and d, the formula is equivalent to a fraction with numerator and denominator parameters.

#### 16.2.11 Example of units dictionary

This dictionary contains definitions corresponding to all the base and derived units defined by in the SI system [SI], plus a selection of conventional units to illustrate the usage of these components.

```
<gml:Dictionary gml:id="unitsDictionary">
<gml:description>A dictionary of units of measure</gml:description>
<gml:name>OWS-1.2 Units</gml:name>
<gml:dictionaryEntry>
<gml:DefinitionCollection gml:id="SIBaseUnits">
<gml:description>The Base Units from the SI units system.</gml:description>
<gml:name>SI Base Units</gml:name>
<gml:dictionaryEntry>
<gml:BaseUnit gml:id="m">
<gml:description>The metre is the length of the path travelled by light in vacuum during a time interval of 1/299 792
458 of a second.</gml:description>
<gml:name codeSpace="http://www.bipm.fr/en/3_SI/base_units.html">metre</gml:name>
<gml:name xml:lang="en/US">meter</gml:name>
<gml:quantityType>length</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI/base_units.html">m</gml:catalogSymbol>
<gml:unitsSystem xlink:href="http://www.bipm.fr/en/3_SI"/>
</gml:BaseUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:BaseUnit gml:id="kg">
<gml:description>The kilogram is the unit of mass; it is equal to the mass of the international prototype of the
kilogram. </gml:description>
<gml:name codeSpace="http://www.bipm.fr/en/3_SI/base_units.html">kilogram</gml:name>
<gml:quantityType>Mass</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI/base_units.html">kg</gml:catalogSymbol>
<gml:unitsSystem xlink:href="http://www.bipm.fr/en/3_SI"/>
</gml:BaseUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:BaseUnit gml:id="s">
```

```

<gml:description>The second is the duration of 9 192 631 770 periods of the radiation corresponding to the transition between the two hyperfine levels of the ground state of the caesium 133 atom.</gml:description>
<gml:name codeSpace="http://www.bipm.fr/en/3_SI/base_units.html">second</gml:name>
<gml:quantityType>Time</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI/base_units.html">s</gml:catalogSymbol>
<gml:unitsSystem xlink:href="http://www.bipm.fr/en/3_SI"/>
</gml:BaseUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:BaseUnit gml:id="A">
<gml:description>The ampere is that constant current which, if maintained in two straight parallel conductors of infinite length, of negligible circular cross-section, and placed 1 metre apart in vacuum, would produce between these conductors a force equal to  $2 \times 10^{-7}$  newton per metre of length.</gml:description>
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">Ampere</gml:name>
<gml:quantityType>Electric current</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">A</gml:catalogSymbol>
<gml:unitsSystem xlink:href="http://www.bipm.fr/en/3_SI"/>
</gml:BaseUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:BaseUnit gml:id="K">
<gml:description>The kelvin, unit of thermodynamic temperature, is the fraction 1/273.16 of the thermodynamic temperature of the triple point of water.</gml:description>
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">kelvin</gml:name>
<gml:quantityType>Thermodynamic temperature</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">K</gml:catalogSymbol>
<gml:unitsSystem xlink:href="http://www.bipm.fr/en/3_SI"/>
</gml:BaseUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:BaseUnit gml:id="mol">
<gml:description>1. The mole is the amount of substance of a system which contains as many elementary entities as there are atoms in 0.012 kilogram of carbon 12.
2. When the mole is used, the elementary entities shall be specified and may be atoms, molecules, ions, electrons, other particles, or specified groups of such particles.</gml:description>
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">mole</gml:name>
<gml:quantityType>Amount of substance</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">mol</gml:catalogSymbol>
<gml:unitsSystem xlink:href="http://www.bipm.fr/en/3_SI"/>
</gml:BaseUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:BaseUnit gml:id="cd">
<gml:description> The candela is the luminous intensity, in a given direction, of a source that emits monochromatic radiation of frequency  $540 \times 10^{12}$  hertz and that has a radiant intensity in that direction of 1/683 watt per steradian.</gml:description>
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">candela</gml:name>
<gml:quantityType>Luminous intensity</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">cd</gml:catalogSymbol>
<gml:unitsSystem xlink:href="http://www.bipm.fr/en/3_SI"/>
</gml:BaseUnit>
</gml:dictionaryEntry>
</gml:DefinitionCollection>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:DefinitionCollection gml:id="SIDerivedUnits">
<gml:description>The Derived Units from the SI units system. These are all derived as a product of SI Base Units, except degrees Celsius in which the conversion formula to the SI Base Unit (kelvin) involves an offset.</gml:description>
<gml:name>SI Derived Units</gml:name>
<gml:dictionaryEntry>
<gml:DerivedUnit gml:id="rad">
<gml:name codeSpace="http://www.bipm.fr/en/3_SI"> radian</gml:name>
<gml:quantityType>plane angle</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">rad</gml:catalogSymbol>

```

```
<gml:derivationUnitTerm uom="#m" exponent="1"/>
<gml:derivationUnitTerm uom="#m" exponent="-1"/>
</gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:DerivedUnit gml:id="sr">
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">steradian</gml:name>
<gml:quantityType>solid angle</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">sr</gml:catalogSymbol>
<gml:derivationUnitTerm uom="#m" exponent="2"/>
<gml:derivationUnitTerm uom="#m" exponent="-2"/>
</gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:DerivedUnit gml:id="Hz">
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">hertz</gml:name>
<gml:quantityType>frequency</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">Hz</gml:catalogSymbol>
<gml:derivationUnitTerm uom="#s" exponent="-1"/>
</gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:DerivedUnit gml:id="N">
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">newton</gml:name>
<gml:quantityType>force</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">N</gml:catalogSymbol>
<gml:derivationUnitTerm uom="#m" exponent="1"/>
<gml:derivationUnitTerm uom="#kg" exponent="1"/>
<gml:derivationUnitTerm uom="#s" exponent="-2"/>
</gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:DerivedUnit gml:id="Pa">
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">pascal</gml:name>
<gml:quantityType>pressure, stress</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">Pa</gml:catalogSymbol>
<gml:derivationUnitTerm uom="#N" exponent="1"/>
<gml:derivationUnitTerm uom="#m" exponent="-2"/>
</gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:DerivedUnit gml:id="J">
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">joule</gml:name>
<gml:quantityType>energy, work, quantity of heat</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">J</gml:catalogSymbol>
<gml:derivationUnitTerm uom="#N" exponent="1"/>
<gml:derivationUnitTerm uom="#m" exponent="1"/>
</gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:DerivedUnit gml:id="W">
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">watt</gml:name>
<gml:quantityType>power, radiant flux</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">W</gml:catalogSymbol>
<gml:derivationUnitTerm uom="#J" exponent="1"/>
<gml:derivationUnitTerm uom="#s" exponent="-1"/>
</gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
<gml:DerivedUnit gml:id="C">
<gml:name codeSpace="http://www.bipm.fr/en/3_SI">coulomb</gml:name>
<gml:quantityType>electric charge, quantity of electricity</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">C</gml:catalogSymbol>
<gml:derivationUnitTerm uom="#s" exponent="1"/>
<gml:derivationUnitTerm uom="#A" exponent="1"/>
</gml:DerivedUnit>
</gml:dictionaryEntry>
```

```

<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="V">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">volt</gml:name>
    <gml:quantityType>electric potential difference, electromotive force</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">V</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#W" exponent="1"/>
    <gml:derivationUnitTerm uom="#A" exponent="-1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="F">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">farad</gml:name>
    <gml:quantityType>capacitance</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">F</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#C" exponent="1"/>
    <gml:derivationUnitTerm uom="#V" exponent="-1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="ohm">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">ohm</gml:name>
    <gml:quantityType>electric resistance</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">? </gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#V" exponent="1"/>
    <gml:derivationUnitTerm uom="#A" exponent="-1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="S">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">siemens</gml:name>
    <gml:quantityType>electric conductance</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">S</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#A" exponent="1"/>
    <gml:derivationUnitTerm uom="#V" exponent="-1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="Wb">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">weber</gml:name>
    <gml:quantityType>magnetic flux</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">Wb</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#V" exponent="1"/>
    <gml:derivationUnitTerm uom="#s" exponent="1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="T">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">tesla</gml:name>
    <gml:quantityType>magnetic flux density</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">T</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#Wb" exponent="1"/>
    <gml:derivationUnitTerm uom="#m" exponent="-2"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="H">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">henry</gml:name>
    <gml:quantityType>inductance</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">H</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#Wb" exponent="1"/>
    <gml:derivationUnitTerm uom="#A" exponent="-1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:ConventionalUnit gml:id="degC">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">degree Celsius</gml:name>

```

```

<gml:quantityType>Celsius temperature</gml:quantityType>
<gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">°C</gml:catalogSymbol>
<gml:conversionToPreferredUnit uom="#K">
  <gml:formula>
    <gml:a>273.16</gml:a>
    <gml:b>1</gml:b>
    <gml:c>1</gml:c>
  </gml:formula>
</gml:conversionToPreferredUnit>
</gml:ConventionalUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="lm">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">lumen</gml:name>
    <gml:quantityType>luminous flux</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">lm</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#cd" exponent="1"/>
    <gml:derivationUnitTerm uom="#sr" exponent="1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="lx">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">lux</gml:name>
    <gml:quantityType>illuminance</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">lx</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#lm" exponent="1"/>
    <gml:derivationUnitTerm uom="#fm" exponent="-2"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="Bq">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">becquerel</gml:name>
    <gml:quantityType>activity (referred to a radionuclide)</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">Bq</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#s" exponent="-1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="Gy">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">gray</gml:name>
    <gml:quantityType>absorbed dose, specific energy (imparted), kerma</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">Gy</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#J" exponent="1"/>
    <gml:derivationUnitTerm uom="#kg" exponent="-1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="Sv">
    <gml:name codeSpace="http://www.bipm.fr/en/3_SI">sievert</gml:name>
    <gml:quantityType>dose equivalent, ambient dose equivalent, directional dose equivalent, personal dose equivalent, organ equivalent dose</gml:quantityType>
    <gml:catalogSymbol codeSpace="http://www.bipm.fr/en/3_SI">Sv</gml:catalogSymbol>
    <gml:derivationUnitTerm uom="#J" exponent="1"/>
    <gml:derivationUnitTerm uom="#kg" exponent="-1"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:DefinitionCollection>
  <gml:dictionaryEntry>
    <gml:DefinitionCollection gml:id="ConventionalUnitsDictionary">
      <gml:description>A collection of Conventional Units. These are units of measure which are either widely used or important within a specific community. For most of these there is
      1. a known derivation from more primitive units, which may or may not be SI Base Units, or
      2. a known conversion to a preferred unit, which may or may not be an SI Base or Derived unit, through rescaling and offset,
      or both. </gml:description>
      <gml:name>Conventional units.</gml:name>
    </gml:DefinitionCollection>
  </gml:dictionaryEntry>
</gml:DefinitionCollection>

```

```

<gml:dictionaryEntry>
  <gml:ConventionalUnit gml:id="GHz">
    <gml:name>GigaHertz</gml:name>
    <gml:quantityType>frequency</gml:quantityType>
    <gml:catalogSymbol>GHz</gml:catalogSymbol>
    <gml:conversionToPreferredUnit uom="#Hz">
      <gml:factor>1.e9</gml:factor>
    </gml:conversionToPreferredUnit>
  </gml:ConventionalUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:ConventionalUnit gml:id="cal">
    <gml:name>Calorie</gml:name>
    <gml:quantityType>energy</gml:quantityType>
    <gml:catalogSymbol>cal</gml:catalogSymbol>
    <gml:conversionToPreferredUnit uom="#J">
      <gml:factor>4.1868</gml:factor>
    </gml:conversionToPreferredUnit>
  </gml:ConventionalUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:DerivedUnit gml:id="m3">
    <gml:name>cubic metre</gml:name>
    <gml:quantityType>Volume</gml:quantityType>
    <gml:derivationUnitTerm uom="#m" exponent="3"/>
  </gml:DerivedUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:ConventionalUnit gml:id="l">
    <gml:name>litre</gml:name>
    <gml:quantityType>Volume</gml:quantityType>
    <gml:conversionToPreferredUnit uom="#m3">
      <gml:factor>0.001</gml:factor>
    </gml:conversionToPreferredUnit>
  </gml:ConventionalUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:ConventionalUnit gml:id="poise">
    <gml:name>poise</gml:name>
    <gml:quantityType>dynamic viscosity</gml:quantityType>
    <gml:catalogSymbol>po</gml:catalogSymbol>
    <gml:conversionToPreferredUnit uom="#Pa.s">
      <gml:factor>0.1</gml:factor>
    </gml:conversionToPreferredUnit>
    <gml:derivationUnitTerm uom="#cm" exponent="-1"/>
    <gml:derivationUnitTerm uom="#g" exponent="1"/>
    <gml:derivationUnitTerm uom="#s" exponent="1"/>
  </gml:ConventionalUnit>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
  <gml:ConventionalUnit gml:id="ft">
    <gml:name>foot</gml:name>
    <gml:quantityType>length</gml:quantityType>
    <gml:catalogSymbol>ft</gml:catalogSymbol>
    <gml:conversionToPreferredUnit uom="#m">
      <gml:factor>0.3048</gml:factor>
    </gml:conversionToPreferredUnit>
  </gml:ConventionalUnit>
</gml:dictionaryEntry>
</gml:DefinitionCollection>
</gml:dictionaryEntry>
</gml:Dictionary>

```

## 16.3 Measures schema

A schema for specific measure types is listed in Annex C; it is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:measures:v3.0

gml:MeasureType is defined in basicTypes.xsd. The measure types defined here correspond with a set of convenience measure types described in ISO 19103. The XML implementation is based on the XML Schema simpleType "double" which supports both decimal and scientific notation, and includes an XML attribute "uom" which refers to the units of measure for the value. Note that, there is no requirement to *store* values using any particular format, and applications receiving elements of this type may choose to coerce the data to any other type as convenient.

### 16.3.1 gml:measure

This is the value of a quantity, together with its units. It is declared as follows:

```
<element name="measure" type="gml:MeasureType"/>
```

In an instance document the element might appear:

```
<gml:measure uom="#m">1.76</gml:measure>
```

The XML attribute **uom** is used to hold a reference to the scale or units by which the amount should be multiplied.

The "uom" attribute uses a URI to refer to a unit of measure definition. The definition may be within the same XML document or external. A definition within the same document would normally use the UnitDefinition element described above, which carries an ID handle as the value of its gml:id attribute. A measure element referring to this definition would normally use the abbreviated Xpointer form of URI.

For convenience the handle on the definition (the value of the gml:id attribute) will normally be mnemonic based on the unit name, such as "m" for metre. So if the definition is in the same document as the reference, the URI may be an abbreviated Xpointer reference [URI] which is presented with the "#" symbol prepended to the handle, i.e. "#m" as in this example.

### 16.3.2 Scalar measure types

A set of specific measure types are defined as vacuous extensions (i.e. aliases) of gml:MeasureType. A prototypical definition is as follows:

```
<complexType name="LengthType">
  <simpleContent>
    <extension base="gml:MeasureType"/>
  </simpleContent>
</complexType>
```

This content model supports the description of a length (or distance) quantity, with its units. The unit of measure referenced by uom shall be suitable for a length, such as metres or feet.

The other measure types that are defined following this pattern are: ScaleType, GridLengthType, AreaType, VolumeType, SpeedType, TimeType.

Elements using these content models might appear in a data instance as follows:

```
<my:length uom="#m">1.76</my:length>
<my:scale uom="#percent">20.</my:scale>
<my:gridLength uom="#pixelSpacing">480</my:gridLength>
<my:gridLength uom="#imageHeight">0.00208333333333</my:gridLength>
<my:area uom="#Ha">1.76</my:area>
```

```

<my:volume uom="#l">0.45</my:volume>
<my:speed uom="#kmph">73.0</my:speed>
<my:time uom="#minutes">30.</my:time>

```

Note that the last element addresses the same functional requirements as the elements in the **gml:\_timeLength** substitution group, defined in clause 14.

### 16.3.3 gml:angle

The **gml:angle** element is used to record the value of an angle quantity as a single number, with its units. **gml:AngleType** is derived trivially from **gml:MeasureType** using the method described above, with the restriction that the unit of measure referenced by **uom** shall be suitable for an angle, such as degrees or radians. It is declared in the schema as follows:

```

<element name="angle" type="gml:AngleType"/>

<complexType name="AngleType">
  <simpleContent>
    <extension base="gml:MeasureType"/>
  </simpleContent>
</complexType>

```

In an instance document the element might appear:

```
<gml:angle uom="#gradians">95.</gml:angle>
```

### 16.3.4 gml:dmsAngle

The **gml:dmsAngle** element is used to record the value of an angle in degree-minute-second or degree-minute format. It uses the following schema declarations:

```

<element name="dmsAngle" type="gml:DMSAngleType"/>

<complexType name="DMSAngleType">
  <sequence>
    <element ref="gml:degrees"/>
    <choice minOccurs="0">
      <element ref="gml:decimalMinutes"/>
      <sequence>
        <element ref="gml:minutes"/>
        <element ref="gml:seconds" minOccurs="0"/>
      </sequence>
    </choice>
  </sequence>
</complexType>

<element name="degrees" type="gml:DegreesType"/>

<complexType name="DegreesType">
  <simpleContent>
    <extension base="gml:DegreeValueType">
      <attribute name="direction">
        <simpleType>
          <restriction base="string">
            <enumeration value="N"/>
            <enumeration value="E"/>
            <enumeration value="S"/>
            <enumeration value="W"/>
            <enumeration value="+"/>
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </simpleContent>
</complexType>

```

```

<enumeration value="-"/>
</restriction>
</simpleType>
</attribute>
</extension>
</simpleContent>
</complexType>

<simpleType name="DegreeValueType">
<restriction base="nonNegativeInteger">
<maxInclusive value="359"/>
</restriction>
</simpleType>

<element name="decimalMinutes" type="gml:DecimalMinutesType"/>

<simpleType name="DecimalMinutesType">
<restriction base="decimal">
<minInclusive value="0.00"/>
<maxExclusive value="60.00"/>
</restriction>
</simpleType>

<element name="minutes" type="gml:ArcMinutesType"/>

<simpleType name="ArcMinutesType">
<restriction base="nonNegativeInteger">
<maxInclusive value="59"/>
</restriction>
</simpleType>

<element name="seconds" type="gml:ArcSecondsType"/>

<simpleType name="ArcSecondsType">
<restriction base="decimal">
<minInclusive value="0.00"/>
<maxExclusive value="60.00"/>
</restriction>
</simpleType>

```

In an instance document the element might appear in one of the following forms:

```

<gml:dmsAngle>
<gml:degrees direction="S">35</gml:degrees>
<gml:minutes>45</gml:minutes>
<gml:seconds>21.98</gml:seconds>
</gml:dmsAngle>

<gml:dmsAngle>
<gml:degrees direction="S">35</gml:degrees>
<gml:decimalMinutes>45.43</gml:decimalMinutes>
</gml:dmsAngle>

```

### 16.3.5 gml:degrees

The **degrees** element allows an integer number of degrees with identification of the angle direction. This element is intended to be used within geographic positions, and has an XML attribute **direction** that can take values

"N" or "S" for Latitude, meaning North or South of the equator;

"E" or "W" for Longitude, meaning East or West of the prime meridian;

"+" or "-" for other angles, in the specified rotational direction from a specified reference direction.

### **16.3.6 `gml:decimalMinutes`**

Decimal number of arc-minutes for use within a degree-minute angular value.

### **16.3.7 `gml:minutes`**

Integer number of arc-minutes for use within a degree-minute-second angular value.

### **16.3.8 `gml:seconds`**

Number of arc-seconds for use within a degree-minute-second angular value.

### **16.3.9 `gml:AngleChoiceType`**

To support the choice of either encoding for angles in a content model, a convenience type `gml:AngleChoiceType` is provided. This element contains another element, either a angle or a dmsAngle. It is declared in the schema as follows:

```
<complexType name="AngleChoiceType">
  <choice>
    <element ref="gml:angle"/>
    <element ref="gml:dmsAngle"/>
  </choice>
</complexType>
```

In an instance document an element of this type might appear:

```
<my:angle>
  <gml:angle uom="#radians">95.</gml:simpleAngle>
</my:angle>

<my:angle>
  <gml:dmsAngle>
    <gml:degrees direction="S">35</gml:degrees>
    <gml:decimalMinutes>45.43</gml:decimalMinutes>
  </gml:dmsAngle>
</my:angle>
```

## **16.4 Value Objects schema**

### **16.4.1 Introduction**

The schema document `valueObjects.xsd` describing components for generic Values is listed in Annex C. It is identified by the following location-independent name (using URN syntax):

`urn:opengis:specification:gml:schema-xsd:valueObjects:v3.0`

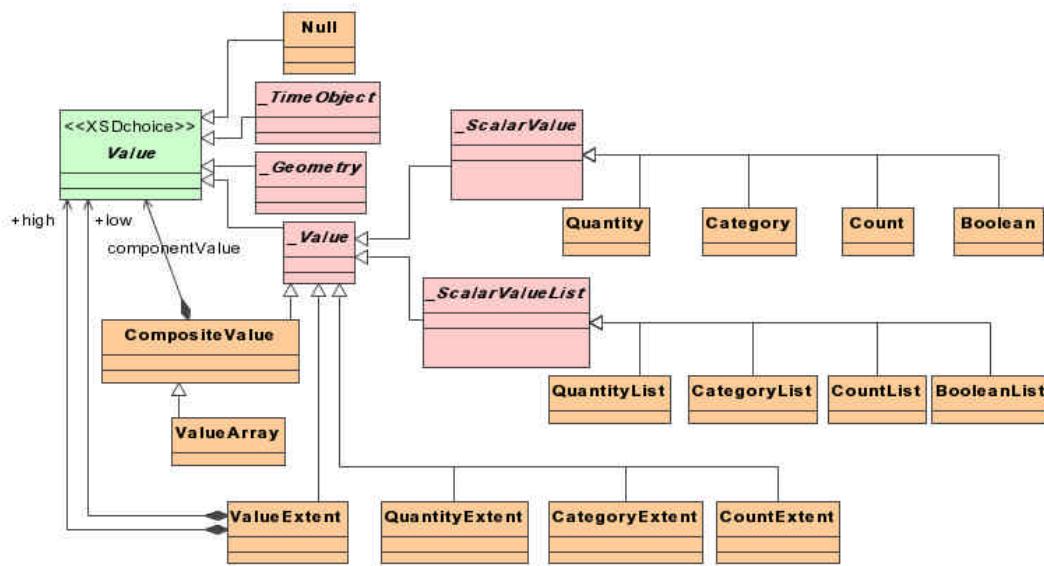
The elements declared in `valueObjects.xsd` build on elements and types from other GML schemas, in particular `_TimeObject` from `temporal.xsd`, `_Geometry` from `geometryBasic0d1d.xsd`, and the following types from `basicTypes.xsd`: `MeasureType`, `MeasureListType`, `CodeType`, `CodeOrNullListType`, `BooleanOrNullListType`, `IntegerOrNullList`.

Of particular interest are elements declared in `valueObjects.xsd` that are the heads of substitution groups, and one named choice group. These are the primary reason for the value objects schema, since they can act as variables in the definition of content models, such as `Observations`, when it is desired to permit alternative value types to occur some of which may have complex content such as arrays, geometry and time objects, and where it is useful not to prescribe the actual value type in advance. The members of the groups include quantities, category classifications, Boolean, count, temporal and spatial values, and aggregates of these.

NOTE The elements declared in this schema are used for **direct representation** of values. Their content models are in general not derived from `gml:AbstractGMLType` and they do not carry an identifier.

#### 16.4.2 Value element hierarchy

The component hierarchy is illustrated in the following UML class diagram. UML generalization relationships are used to indicate XML Schema substitution group and choice group membership. UML composition relationships are used to indicate membership in an XML Schema type content model.



**Figure 16.4.1 Substitution groups and composition relationships in the value schema**

The following relationships are defined in the `valueObjects.xsd` schema:

- Concrete elements `Quantity`, `Category`, `Count` and `Boolean` are substitutable for the abstract element `_ScalarValue`.
- Concrete elements `QuantityList`, `CategoryList`, `CountList` and `BooleanList` are substitutable for the abstract element `_ScalarValueList`.
- Concrete element `ValueArray` is substitutable for the concrete element `CompositeValue`.
- Abstract elements `_ScalarValue` and `_ScalarValueList`, and concrete elements `CompositeValue`, `ValueExtent`, `CategoryExtent`, `CountExtent` and `QuantityExtent` are substitutable for abstract element `_Value`.
- Abstract elements `_Value`, `_TimeObject` (from `temporal.xsd`) and `_Geometry` (from `geometryBasic0d1d.xsd`), and concrete element `Null` (from `gmlBase.xsd`) are all in a choice group named `Value`, which is used for compositing in `CompositeValue` and `ValueExtent`.
- Schemas which need values may use the abstract element `_Value` in a content model in order to permit any of the `_ScalarValue`'s, `_ScalarValueList`'s, `CompositeValue` or `ValueExtent` to occur in an instance, or the named group `Value` to also permit `TimeObject`'s, `Geometry`'s, and `Null`'s.

#### 16.4.3 gml:Boolean, gml:BooleanList

For recording a value or list of values from two-valued logic, using the XML Schema `boolean` type. These elements use the following schema declarations:

```
<element name="Boolean" type="boolean" substitutionGroup="gml:_ScalarValue"/>  
<element name="BooleanList" type="gml:booleanOrNullList" substitutionGroup="gml:_ScalarValueList"/>
```

These elements are described in clause 7.3.4.1.

In an instance the following examples may be found:

```
<gml:Boolean>1</gml:Boolean>  
<gml:Boolean>false</gml:Boolean>  
<gml:BooleanList>1 missing 0 1 1 http://my.big.org/explanations/theDogAtelt 0 1</gml:BooleanList>
```

These examples illustrate the use of the various Boolean values {1, 0, true, false} and also the fact that null values such as "missing" or a URI can be embedded within a list.

#### 16.4.4 gml:Category, gml:CategoryList

For recording terms representing a classification. These elements use the following schema declarations:

```
<element name="Category" type="gml:CodeType" substitutionGroup="gml:_ScalarValue"/>  
<element name="CategoryList" type="gml:CodeOrNullListType" substitutionGroup="gml:_ScalarValueList"/>
```

A Category has an optional XML attribute **codeSpace**, whose value is a URI which identifies a dictionary, codelist or authority for the term. In an instance the following examples may be found:

```
<gml:Category>good</gml:Category>  
<gml:Category codeSpace="http://my.big.org/dictionaries/rocktypes">Syenite</gml:Category>  
<gml:CategoryList codeSpace="http://my.big.org/dictionaries/rocktypes">Syenite Granite missing Tuff</gml:CategoryList>  
<gml:CategoryList codeSpace="http://my.big.org/species">bettong numbat phasogale wallaby possum</gml:CategoryList>
```

#### 16.4.5 gml:Count, gml:CountList

For recording integers representing a rate of occurrence. These elements use the following schema declarations:

```
<element name="Count" type="integer" substitutionGroup="gml:_ScalarValue"/>  
<element name="CountList" type="gml:integerOrNullList" substitutionGroup="gml:_ScalarValueList"/>
```

In an instance the following examples may be found:

```
<gml:Count>513</gml:Count>  
<gml:CountList>34 56 2 inapplicable 153</gml:CountList>
```

#### 16.4.6 gml:Quantity, gml:QuantityList

For recording numeric values with a scale. The content of the element is an amount using the XML Schema type double which permits decimal or scientific notation. These elements use the following schema declarations:

```
<element name="Quantity" type="gml:MeasureType" substitutionGroup="gml:_ScalarValue"/>  
<element name="QuantityList" type="gml:MeasureOrNullListType" substitutionGroup="gml:_ScalarValueList"/>
```

An XML attribute **uom** ("unit of measure") is required, whose value is a URI which identifies the definition of a ratio scale or units by which the numeric value shall be multiplied, or an interval or position scale on which the value occurs. In an instance the following examples may be found:

```
<gml:Quantity uom="#m">4.32e-4</gml:Quantity>
<gml:QuantityList uom="#C">21. 37. withheld 25.</gml:QuantityList>
```

#### 16.4.7 gml:\_Value, gml:\_ScalarValue, gml:\_ScalarValueList

**Value** is an abstract element which acts as the head of a substitution group which contains **\_ScalarValue**, **\_ScalarValueList**, **CompositeValue** and **ValueExtent**, and (transitively) the elements in their substitution groups.

**ScalarValue** is an abstract element which acts as the head of a substitution group which contains **Boolean**, **Category**, **Count** and **Quantity**, and (transitively) the elements in their substitution groups.

**ScalarValueList** is an abstract element which acts as the head of a substitution group which contains **BooleanList**, **CategoryList**, **CountList** and **QuantityList**, and (transitively) the elements in their substitution groups.

These elements use the following schema declarations:

```
<element name="_Value" abstract="true" substitutionGroup="gml:_Object">
<element name="_ScalarValue" abstract="true" substitutionGroup="gml:_Value"/>
<element name="_ScalarValueList" abstract="true" substitutionGroup="gml:_Value"/>
```

These elements may be used in an application schema as variables, so that in an XML instance document any member of its substitution group may occur.

#### 16.4.8 gml:Value

This is a convenience choice group which unifies generic Values defined in this schema document with Geometry and Temporal objects and the Measures described above, so that any of these may be used within aggregate Values. This element uses the following schema declaration:

```
<group name="Value">
<choice>
<element ref="gml:_Value"/>
<element ref="gml:_Geometry"/>
<element ref="gml:_TimeObject"/>
<element ref="gml:Null"/>
<element ref="gml:measure"/>
</choice>
</group>
```

#### 16.4.9 gml:valueProperty, gml:valueComponent, gml:valueComponents

Elements that instantiates a GML property which refers to, or contains, a Value or Values. These elements use the following schema declarations:

```
<element name="valueProperty" type="gml:ValuePropertyType"/>
<element name="valueComponent" type="gml:ValuePropertyType"/>

<complexType name="ValuePropertyType">
<sequence minOccurs="0">
<group ref="gml:Value"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

```

<element name="valueComponents" type="gml:ValueArrayType"/>

<complexType name="ValueArrayPropertyType">
  <sequence>
    <group ref="gml:Value" maxOccurs="unbounded"/>
  </sequence>
</complexType>

```

Note that both ValuePropertyType and ValueArrayPropertyType have the group named “Value” as their content. This means that any of the elements in the Value choice group, or in the substitution groups of the members of the choice group can occur as the content of a valueProperty.

The valueProperty element is a convenience element for general use. The valueComponent and valueComponents elements are specifically used in compositing.

#### 16.4.10 gml:CompositeValue

CompositeValue is an aggregate value built from other Values using the Composite pattern. It contains zero or an arbitrary number of valueComponent elements, and zero or one valueComponents elements. It may be used for strongly coupled aggregates (vectors, tensors) or for arbitrary collections of values. This element uses the following schema declarations:

```

<element name="CompositeValue" type="gml:CompositeValueType" substitutionGroup="gml:_Value"/>

<complexType name="CompositeValueType">
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <sequence>
        <element ref="gml:valueComponent" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:valueComponents" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

In an instance a CompositeValue may appear as in the following examples:

```

<gml:CompositeValue>
  <gml:valueComponent>
    <gml:QuantityList uom="#C">21. 37. withheld 25.</gml:QuantityList>
  </gml:valueComponent>
  <gml:valueComponent>
    <gml:Category>good</gml:Category>
  </gml:valueComponent>
  <gml:valueComponent>
    <gml:Null>template</gml:Null>
  </gml:valueComponent>
  <gml:valueComponents>
    <gml:Point srsName="urn:EPSG:geographicCRS:4326"><gml:pos>171. -32.</gml:pos></gml:Point>
    <gml:Point srsName="urn:EPSG:geographicCRS:4326"><gml:pos>170. -35.</gml:pos></gml:Point>
    <gml:Point srsName="urn:EPSG:geographicCRS:4326"><gml:pos>174. -37.</gml:pos></gml:Point>
  </gml:valueComponents>
</gml:CompositeValue>

```

```

<gml:CompositeValue>
  <gml:valueComponents>
    <gml:Point srsName="urn:EPSG:geographicCRS:4326"><gml:pos>-67.563 -13.834</gml:pos></gml:Point>
    <gml:Quantity uom="#km">632.</gml:Quantity>
    <gml:TimeInstant><gml:timePosition>1994-06-09T00:33:16.4</gml:timePosition></gml:TimeInstant>
    <gml:Quantity uom="#mom">-1.00</gml:Quantity>
    <gml:Quantity uom="#mom">0.92</gml:Quantity>
    <gml:Quantity uom="#mom">0.09</gml:Quantity>
  </gml:valueComponents>
</gml:CompositeValue>

```

```
<gml:Quantity uom="#mom">-1.69</gml:Quantity>
<gml:Quantity uom="#mom">-0.09</gml:Quantity>
<gml:Quantity uom="#mom">-0.37</gml:Quantity>
</gml:valueComponents>
</gml:CompositeValue>
```

#### 16.4.11 gml:ValueArray

A Value Array is used for homogeneous arrays of primitive and aggregate values.

The member values may be scalars, composites, arrays or lists. This element uses the following schema declarations:

```
<element name="ValueArray" type="gml:ValueArrayType" substitutionGroup="gml:CompositeValue">
  <annotation>
    <appinfo>
      <sch:pattern name="Check either codeSpace or uom not both">
        <sch:rule context="gml:ValueArray">
          <sch:report test="@codeSpace and @uom">ValueArray may not carry both a reference to a codeSpace and a uom</sch:report>
        </sch:rule>
      </sch:pattern>
      <sch:pattern name="Check components are homogeneous">
        <sch:rule context="gml:ValueArray">
          <sch:assert test="count(gml:valueComponent/*) = count(gml:valueComponent/*[name() = name(../.gml:valueComponent[1]/*[1])])">All components of <sch:name/> shall be of the same type</sch:assert>
          <sch:assert test="count(gml:valueComponents/*) = count(gml:valueComponents/*[name() = name(../*[1])])">All components of <sch:name/> shall be of the same type</sch:assert>
        </sch:rule>
      </sch:pattern>
    </appinfo>
  </annotation>
</element>

<complexType name="ValueArrayType">
  <complexContent>
    <extension base="gml:CompositeValueType">
      <attributeGroup ref="gml:referenceSystem"/>
    </extension>
  </complexContent>
</complexType>

<attributeGroup name="referenceSystem">
  <attribute name="codeSpace" type="anyURI" use="optional"/>
  <attribute name="uom" type="anyURI" use="optional"/>
</attributeGroup>
```

ValueArray has the same content model as CompositeValue, but the member values shall be homogeneous. The element declaration contains a Schematron constraint which expresses this restriction precisely. Since the members are homogeneous, the referenceSystem (**uom**, **codeSpace**) may be specified on the ValueArray itself and inherited by all the members if desired.

The ValueArray element may appear in instances as follows.

In the first example a set of Points are each the value of a valueComponent property. One of the values is provided by-reference, using the standard xlink:href syntax:

```
<gml:ValueArray>
  <gml:valueComponent>
    <gml:Point srsName="urn:EPSG:geographicCRS:63266405">
      <gml:pos>-32. 171.</gml:pos>
    </gml:Point>
  </gml:valueComponent>
  <gml:valueComponent>
```

```

<gml:Point srsName="urn:EPSG:geographicCRS: 63266405">
  <gml:pos>-35. 170.</gml:pos>
</gml:Point>
</gml:valueComponent>
<gml:valueComponent xlink:href="http://my.big.org/locations/points/point456"/>
</gml:ValueArray>
```

In the second example a set of Quantities are contained within a valueComponents property. One of the values is not available, indicated by a Null:

```

<gml:ValueArray>
  <gml:valueComponents>
    <gml:Quantity uom="#C">21.</gml:Quantity>
    <gml:Quantity uom="#C">37.</gml:Quantity>
    <gml:Null>missing</gml:Null>
  </gml:valueComponents>
</gml:ValueArray>
```

Note that a \_ScalarValueList is usually preferred for arrays of Scalar Values since this is a more efficient encoding. The information in the previous example can be expressed:

```
<gml:QuantityList uom="#C">21. 37. missing</gml:QuantityList>
```

However, if the values of the components are not scalars, then the explicit form is required.

#### 16.4.12 Typed ValueExtents: gml:CategoryExtent, gml:CountExtent, gml:QuantityExtent

Three elements are provided for typed value extents, for categories, counts and quantities. Their content models are defined by restricting the relevant scalar list types to contain exactly two items as follows:

```

<element name="CategoryExtent" type="gml:CategoryExtentType" substitutionGroup="gml:_Value"/>

<complexType name="CategoryExtentType">
  <simpleContent>
    <restriction base="gml:CodeOrNullListType">
      <length value="2"/>
    </restriction>
  </simpleContent>
</complexType>

<element name="CountExtent" type="gml:CountExtentType" substitutionGroup="gml:_Value"/>

<simpleType name="CountExtentType">
  <restriction base="gml:integerOrNullList">
    <length value="2"/>
  </restriction>
</simpleType>

<element name="QuantityExtent" type="gml:QuantityExtentType" substitutionGroup="gml:_Value"/>

<complexType name="QuantityExtentType">
  <simpleContent>
    <restriction base="gml:MeasureOrNullListType">
      <length value="2"/>
    </restriction>
  </simpleContent>
</complexType>
```

A gml:QuantityExtent element or another element using this type will contain two values and a scale as follows:

```
<gml:QuantityExtent uom="#mm">0. 9.5</gml:QuantityExtent>
```

An element of `gml:CategoryExtentType` is useful if the codeSpace defines a set of ordered terms, for example:

```
<my:AgeRange codeSpace="http://iugg.org/geologicalTimePeriods">Cambrian Devonian</my:AgeRange>
```

Any value extent may describe a single-ended interval by using a Null value for one of the limits, for example:

```
<gml:CountExtent>53 inapplicable</gml:CountExtent>
```

describes the integers starting with 53.

#### **16.4.13 `gml:BooleanPropertyType`, `gml:CategoryPropertyType`, `gml:CountPropertyType`, `gml:QuantityPropertyType`**

A set of convenience types are provided for properties whose content is a specific member of the `gml:_ScalarValue` substitution group. Their definitions follow the same pattern, as exemplified by the definition of `gml:BooleanPropertyType`:

```
<complexType name="BooleanPropertyType">
  <complexContent>
    <restriction base="gml:ValuePropertyType">
      <sequence>
        <element ref="gml:Boolean" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
```

## **17 GML schemas – directions**

### **17.1 Direction schema**

The direction schema, `direction.xsd` provides the GML Application Schema developer with a standard property element to describe direction, and associated objects that can be used to express orientation, direction, heading, bearing or other directional aspects of geographic features. The schema is listed in Annex C. It is identified by the following location-independent name (using URN syntax):

`urn:opengis:specification:gml:schema-xsd:direction:v3.1.0`

#### **17.1.1 `gml:direction`, `gml:DirectionPropertyType`**

The property `gml:direction` is intended as a property to be assigned to features defined in a GML application schema. It is declared in the schema as follows:

```
<element name="direction" type="gml:DirectionPropertyType"/>

<complexType name="DirectionPropertyType">
  <annotation>
    <documentation/>
  </annotation>
  <choice>
    <element ref="gml:DirectionVector"/>
    <element ref="gml:CompassPoint"/>
    <element name="DirectionKeyword" type="gml:CodeType"/>
    <element name="DirectionString" type="gml:StringOrRefType"/>
  </choice>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

The various kinds of direction specifications follow.

### 17.1.2 gml:DirectionVector, gml:DirectionVectorType

Director vectors are specified by providing components of a vector or two angles as follows:

```
<element name="DirectionVector" type="gml:DirectionVectorType"/>

<complexType name="DirectionVectorType">
  <choice>
    <element ref="gml:vector"/>
    <sequence>
      <element name="horizontalAngle" type="gml:AngleType"/>
      <element name="verticalAngle" type="gml:AngleType"/>
    </sequence>
  </choice>
</complexType>
```

The gml:vector element is described in clause 9.1.4.1. This form may appear in an data instance as follows:

```
<gml:direction>
  <gml:DirectionVector>
    <gml:horizontalAngle uom="#degreesEastOfNorth">45.0</gml:horizontalAngle>
    <gml:verticalAngle uom="#degreesDown">0.0</gml:verticalAngle>
  </gml:DirectionVector>
</gml:direction>
```

Angles are specified via gml:AngleType whose content model is defined in clause 16.3.3. This form may appear in an data instance as follows:

```
<gml:direction>
  <gml:DirectionVector>
    <gml:horizontalAngle uom="#degreesEastOfNorth">45.0</gml:horizontalAngle>
    <gml:verticalAngle uom="#degreesDown">0.0</gml:verticalAngle>
  </gml:DirectionVector>
</gml:direction>
```

### 17.1.3 gml:CompassPoint, gml:CompassPointEnumeration

A compass point is specified by a simple enumeration string type. The gml:CompassPoint element is declared in the schema as follows:

```
<element name="CompassPoint" type="gml:CompassPointEnumeration"/>

<simpleType name="CompassPointEnumeration">
  <restriction base="string">
    <enumeration value="N"/>
    <enumeration value="NNE"/>
    <enumeration value="NE"/>
    <enumeration value="ENE"/>
    <enumeration value="E"/>
    <enumeration value="ESE"/>
    <enumeration value="SE"/>
    <enumeration value="SSE"/>
    <enumeration value="S"/>
    <enumeration value="SSW"/>
    <enumeration value="SW"/>
    <enumeration value="WSW"/>
    <enumeration value="W"/>
    <enumeration value="WNW"/>
    <enumeration value="NW"/>
    <enumeration value="NNW"/>
  </restriction>
</simpleType>
```

These directions are necessarily approximate, giving direction with a precision of 22.5°. It is thus generally unnecessary to specify the reference frame, though this may be detailed in the definition of a GML application language.

This form may appear in an data instance as follows:

```
<gml:direction>
  <gml:CompassPoint>WNW</gml:CompassPoint>
</gml:direction>
```

#### 17.1.4 Text Based Directions: gml:DirectionKeyword, gml:DirectionString

Two elements to contain text-based descriptions of direction are provided.

If the direction is specified using a term from a list, gml:KeyWord should be used, and the list indicated using the value of the codeSpace attribute. This form may appear in an data instance as follows:

```
<gml:direction>
  <gml:DirectionKeyword codeSpace="http://my.big.org/terms/direction">onshore</gml:DirectionKeyword>
</gml:direction>
```

If the direction is described in prose, gml:DirectionString should be used, allowing the value to be included inline or by reference. This form may appear in an data instance as follows:

```
<gml:direction>
  <gml:DirectionString>Towards the lighthouse</gml:DirectionString>
</gml:direction>
```

```
<gml:direction>
  <gml:DirectionString xlink:href="http://my.big.org/logbook/20021127/paragraph6"/>
</gml:direction>
```

## 18 GML schemas – observations

### 18.1 Observations

A GML observation models the act of observing, often with a camera, a person or some form of instrument. An observation feature describes the “metadata” associated with an information capture event, together with a value for the result of the observation. This covers a broad range of cases, from a tourist photo (not the photo but the act of taking the photo), to images acquired by space borne sensors or the measurement of a temperature 5 meters below the surfaces of a Lake. See also Clause 7.9. (measures.xsd).

The basic structures introduced in this schema are intended to serve as the foundation for more comprehensive schemas for scientific, technical and engineering measurement schemas.

### 18.2 Observation schema

Observations are described in the schema, observations.xsd. The schema is listed in Annex C. It is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:observation:v3.1.0

This schema describes two kinds of observations, gml:Observation and gml:DirectedObservation.

#### 18.2.1 gml:Observation

The gml:Observation element is declared in the schema as follows:

```

<element name="Observation" type="gml:ObservationType" substitutionGroup="gml:_Feature"/>

<complexType name="ObservationType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:validTime"/>
        <element ref="gml:using" minOccurs="0"/>
        <element ref="gml:target" minOccurs="0"/>
        <element ref="gml:resultOf"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

The content model is a straightforward extension of gml:AbstractFeatureType it automatically has gml:metadataProperty, gml:description, gml:name, gml:location and gml:boundedBy properties.

The gml:validTime element is declared in clause 14.2.2.3. In this context it describes the time of the observation. Note that this can be a time instant or a time period.

### 18.2.2 gml:using

The gml:using property contains or points to a description of a sensor, instrument or procedure used for the observation. It is declared in the schema as follows:

```
<element name="using" type="gml:Feature.PropertyType"/>
```

### 18.2.3 gml:target

The gml:target property contains or points to the specimen, region or station which is the object of the observation. This property element is declared in the schema as follows:

```

<element name="target" type="gml:Target.PropertyType"/>

<element name="subject" type="gml:Target.PropertyType" substitutionGroup="gml:target">

<complexType name="Target.PropertyType">
  <annotation>
    <documentation>Container for an object representing the target or subject of an observation.</documentation>
  </annotation>
  <choice minOccurs="0">
    <element ref="gml:_Feature"/>
    <element ref="gml:_Geometry"/>
  </choice>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

```

This property is particularly useful for remote observations, such as photographs, where the gml:location property might apply to the location of the camera or the location of the field of view, and thus may be ambiguous.

The subject element is provided as a convenient synonym for target. This is the term commonly used in photography.

#### 18.2.4 **gml:resultOf**

The gml:resultOf property indicates the result of the observation. Following the standard GML property pattern (clause ), the value may be inline, or a reference to a value elsewhere. If the value is inline, it must be a member of the gml:\_Object substitution group. It is declared in the schema as follows:

```
<element name="resultOf" type="gml:AssociationType"/>
```

### 18.3 Examples

#### 18.3.1 **gml:Observation**

Some examples of simple observations are as follows:

```
<gml:Observation>
  <gml:location xlink:href="http://www.environment.org/stations/l456"/>
  <gml:validTime>
    <gml:TimeInstant>
      <gml:timePosition>2002-11-12T09:12:00</gml:timePosition>
    </gml:TimeInstant>
  </gml:validTime>
  <gml:using xlink:href="http://www.my.org/sensors/thermometer4"/>
  <gml:target xlink:href="http://www.environment.org/stations/l456"/>
  <gml:resultOf>
    <gml:Quantity uom="#C">18.4</gml:Quantity>
  </gml:resultOf>
</gml:Observation>
```

```
<gml:Observation>
  <gml:location xlink:href="http://www.tourist.org/lookouts/platform4"/>
  <gml:validTime>
    <gml:TimeInstant>
      <gml:timePosition>2002-11-12T09:12:00</gml:timePosition>
    </gml:TimeInstant>
  </gml:validTime>
  <gml:using xlink:href="http://www.my.org/cameras/leica2"/>
  <gml:subject xlink:href="http://www.tourist.org/sights/mountain3"/>
  <gml:resultOf xlink:href="http://www.my.org/photos/landscape1.jpg"/>
</gml:Observation>
```

```
<gml:Observation>
  <gml:location>
    <gml:LocationString>Home</gml:LocationString>
  </gml:location>
  <gml:validTime>
    <gml:TimeInstant>
      <gml:timePosition>2002-10-25T11:37:25</gml:timePosition>
    </gml:TimeInstant>
  </gml:validTime>
  <gml:subject xlink:href="http://www.people.org/kids/abby"/>
  <gml:resultOf xlink:href="myDaughtersPortrait.jpg"/>
</gml:Observation>
```

#### 18.3.2 **gml:DirectedObservation**

A DirectedObservation is the same as an observation except that it adds an additional direction property. This is the direction in which the observation was acquired. Clearly this applies only to certain types of observations such as visual observations by people, or observations obtained from terrestrial cameras.

```
<element name="DirectedObservation" type="gml:DirectedObservationType"
  substitutionGroup="gml: Observation"/>

<complexType name="DirectedObservationType">
```

```

<complexContent>
  <extension base="gml:ObservationType">
    <sequence>
      <element ref="gml:direction"/>
    </sequence>
  </extension>
</complexContent>
</complexType>

```

EXAMPLE      DirectedObservation:

```

<gml:DirectedObservation>
  <gml:validTime>
    <gml:TimeInstant>
      <gml:timePosition>2002-11-12T09:12:00</gml:timePosition>
    </gml:TimeInstant>
  </gml:validTime>
  <gml:using xlink:href="http://www.my.org/cameras/leica2"/>
  <gml:target xlink:href="http://www.tourist.org/sights/mountain3"/>
  <gml:resultOf xlink:href="http://www.my.org/photos/landscape1.jpg"/>
  <gml:direction>
    <gml:CompassPoint>NW</gml:CompassPoint>
  </gml:direction>
</gml:DirectedObservation>

```

### 18.3.3 gml:DirectedObservationAtDistance

A DirectedObservationAtDistance adds an additional distance property. This is the distance from the observer to the subject of the observation. Clearly this applies only to certain types of observations such as visual observations by people, or observations obtained from terrestrial cameras.

```

<element name="DirectedObservationAtDistance" type="gml:DirectedObservationAtDistanceType"
  substitutionGroup="gml: DirectedObservation "/>

<complexType name="DirectedObservationAtDistanceType">
  <complexContent>
    <extension base="gml:DirectedObservationType">
      <sequence>
        <element name="distance" type="gml:MeasureType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

EXAMPLE      DirectedObservationAtDistance:

```

<gml:DirectedObservationAtDistance>
  <gml:validTime>
    <gml:TimeInstant>
      <gml:timePosition>2002-11-12T09:12:00</gml:timePosition>
    </gml:TimeInstant>
  </gml:validTime>
  <gml:using xlink:href="http://www.my.org/cameras/leica2"/>
  <gml:subject xlink:href="http://www.tourist.org/sights/mountain3"/>
  <gml:resultOf xlink:href="http://www.my.org/photos/landscape1.jpg"/>
  <gml:direction>
    <gml:CompassPoint>NW</gml:CompassPoint>
  </gml:direction>
  <gml:distance uom="#m">16500.</gml:distance>
</gml:DirectedObservationAtDistance>

```

## 19 GML schemas – coverages

### 19.1 The coverage model and representations

#### 19.1.1 General Remarks

This clause defines the GML encoding for coverages and is in agreement with the conceptual model outlined in ISO 19123 and in the OGC Abstract Specification, Topic 6.

ISO 19123 provides a definition:

*Coverages support mapping from a spatiotemporal domain to attribute values where attribute types are common to all geographic positions within the spatiotemporal domain. A spatiotemporal domain consists of a collection of direct positions in a coordinate space. Examples of coverages include rasters, triangulated irregular networks, point coverages, and polygon coverages. Coverages are the prevailing data structures in a number of application areas, such as remote sensing, meteorology, and bathymetric, elevation, soil, and vegetation mapping.*

The information describing a coverage is conventionally represented in one of two ways:

- a. As a set of discrete location-value pairs.
- b. As a description of the spatio-temporal domain (multi-geometry, grid) and a description of the set of values from the range, together with a method or rule (which may be implicit) that assigns a value from the range set to each position within the domain.

The first method only applies to domains that are partitioned into discrete components. This representation may be realised in GML as a **homogeneous feature collection** (i.e. all the features have the same set of properties), where the set of locations from the features compose the domain (remember: `gml:location` may refer to any geometry, not just points), and the set of property values compose the range. The mapping from domain to range is trivial: the properties on each feature are assigned to the location of that feature. For coverages whose domain is composed of a large set of locations this explicit representation may, however, be bulky.

The second method is more flexible in a number of ways.

- a) Since the domain and range are homogeneous sets, there may be efficiencies in the representation of either or both domain and range
- b) The values in the range may be represented in analytic form rather than as discrete explicit values, which is also related to the fact that
- c) When the attribute values vary continuously across the domain, a functional form covering the complete domain is required to be able to provide values of the range at arbitrary locations. The function typically involves interpolation, possibly using a process model.

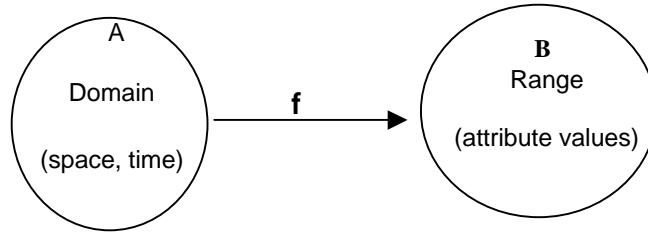
The first representation is typically used during data collection where a set of properties relating to a single location are managed together, or update of a datastore where only a small number of features are manipulated at one time. The second representation is more suitable for analysis, where spatio-temporal patterns and anomalies within a specific property are of interest.

It is the second method, using a functional map over the whole domain, that is the subject of the GML coverage encoding.

#### 19.1.2 Formal description of a coverage

A coverage incorporates a mapping from a spatiotemporal domain to a range set, the latter providing the set in which the attribute values live. The range set can be an arbitrary set including discrete lists, integer or

floating point ranges, and multi-dimensional vector spaces. This conceptual model of a coverage is described in Figure 4.



**Figure 4 – Conceptual Model of a Coverage**

A coverage can be viewed as the graph of the coverage function  $f:A \rightarrow B$ , that is as the set of ordered pairs

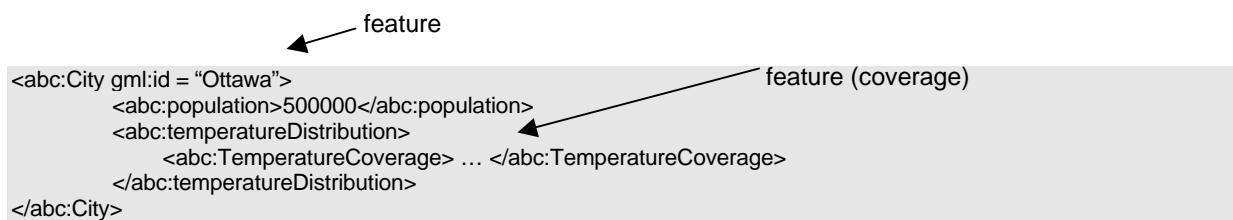
$$\{(x, f(x)) \mid \text{where } x \text{ is in } A\}$$

This view is especially applicable to the GML encoding of a coverage. In the case of a discrete coverage, the domain set  $A$  is partitioned into a collection of subsets (typically a disjoint collection)  $A = \cup A_i$  and the function  $f$  is constant on each  $A_i$ . For a spatial domain, the  $A_i$  are geometry elements, hence the coverage can be viewed as a collection of (geometry,value) pairs, where the value is an element of the range set. If the spatial domain  $A$  is a topological space then the coverage can be viewed as a collection of (topology,value) pairs, where the topology element in the pair is a topological n-chain (in GML terms this is a `gml:TopoPoint`, `gml:TopoCurve`, `gml:TopoSurface` or `gml:TopoSolid`).

#### 19.1.3 Coverage in GML

A coverage is implemented as a GML feature. We can thus speak of a “temperature distribution feature”, or a “remotely sensed image feature”, or a “soil distribution feature”.

As is the case for any GML object, a coverage object may also be the value of a property of a feature. For example, the temperature distribution might be a property of a city feature, so a description of the city of Ottawa might be represented in GML as:



Coverages in GML 3 are supported by two schemas,

- `coverage.xsd`
- `grids.xsd`.

`Coverages.xsd` provides the basic GML 3 coverage model. `Grids.xsd` provides grid geometry structures that are used in the description of gridded coverages but which could be employed for other applications.

Future releases of GML will provide other geometries and temporal complexes for use in coverages.

## 19.2 Grids schema

An implicit description of geometry is one in which the items of the geometry do not explicitly appear in the encoding. Instead, a compact notation records a set of parameters, and a set of objects may be generated using a rule with these parameters.

The schema grids.xsd provides some grid geometries that are used in the description of gridded coverages and other applications. The schema is listed in Annex C. It is identified by the following location-independent name (using URN syntax):

```
urn:opengis:specification:gml:schema-xsd:grids:v3.1.0
```

In GML 3 two grid structures are defined, namely gml:Grid and gml:RectifiedGrid.

### 19.2.1 Unrectified Grid (gml:Grid)

The gml:Grid element is defined in the schema as follows:

```
<element name="Grid" type="gml:GridType" substitutionGroup="gml:_ImplicitGeometry"/>

<complexType name="GridType">
  <complexContent>
    <extension base="gml:AbstractGeometryType">
      <sequence>
        <element name="limits" type="gml:GridLimitsType"/>
        <element name="axisName" type="string" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

The gml:Grid implicitly defines an unrectified grid, which is a network composed of two or more sets of equally spaced parallel lines in which the members of each set intersect each other. The region of interest within the grid is given in terms of its **limits**, being the grid coordinates of diagonally opposed corners of a rectangular region. An **axisName** is provided for each of the axes of the grid. The standard SRSReferenceGroup is inherited from gml:AbstractGeometryType. This includes the **dimension** attribute.

In GML3 the **limits** element contains a single GridEnvelope, according to the following schema definitions:

```
<complexType name="GridLimitsType">
  <sequence>
    <element name="GridEnvelope" type="gml:GridEnvelopeType"/>
  </sequence>
</complexType>

<complexType name="GridEnvelopeType">
  <sequence>
    <element name="low" type="gml:integerList"/>
    <element name="high" type="gml:integerList"/>
  </sequence>
</complexType>
```

The low and high elements are each integerLists, which are coordinate tuples, the coordinates being measured as offsets from the origin of the grid along each axis, of the diagonally opposing corners of a “rectangular” region of interest.

The following example illustrates a simple Grid.

```
<gml:Grid dimension="2">
  <gml:limits>
    <gml:GridEnvelope>
      <gml:low>0 0</gml:low>
      <gml:high>3 3</gml:high>
```

```

</gml:GridEnvelope>
</gml:limits>
<gml:axisName>x</gml:axisName>
<gml:axisName>y</gml:axisName>
</gml:Grid>

```

In this example the Grid has posts (points) at locations (0,0), (0,1),(1,0),(1,1) through to (4,4).

### 19.2.2 Rectified Grid (gml:RectifiedGrid)

A rectified grid is a kind of grid in which the points of the grid have geometric locations. It is defined by specifying the position (in some geometric space) of the grid “origin” and of the vectors that specify the post locations. The RectifiedGrid element is declared in the schema as follows:

```

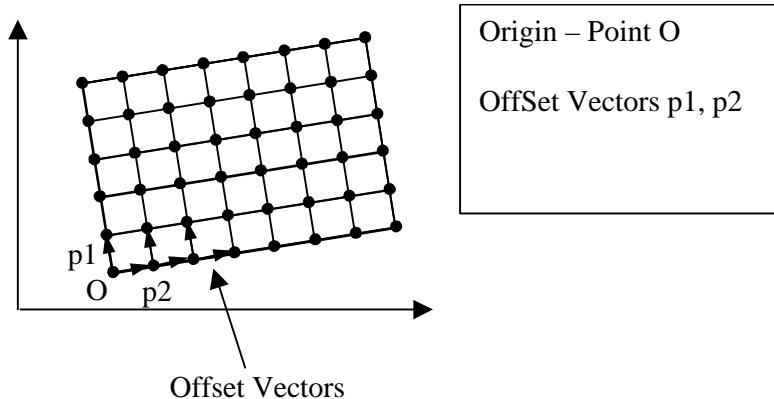
<element name="RectifiedGrid" type="gml:RectifiedGridType" substitutionGroup="gml:Grid"/>

<complexType name="RectifiedGridType">
  <complexContent>
    <extension base="gml:GridType">
      <sequence>
        <element name="origin" type="gml:PointPropertyType"/>
        <element name="offsetVector" type="gml:VectorType" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

Note that the grid limits (post indexes) and axis name properties are inherited from the gml:GridType and that the gml:RectifiedGrid adds an origin (contains a gml:Point) and a set of offset vectors specified using gml:VectorType as described in clause 9.1.4.1.

Figure 5 shows the geometry of the RectifiedGrid.



**Figure 5 – RectifiedGrid Geometry**

An example RectifiedGrid is as follows:

```

<gml:RectifiedGrid dimension="2">
  <gml:limits>
    <gml:GridEnvelope>
      <gml:low>1 1</gml:low>
      <gml:high>3 3</gml:high>
    </gml:GridEnvelope>
  </gml:limits>

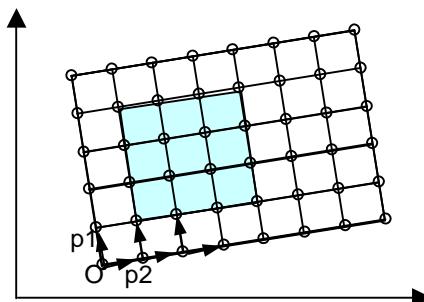
```

```

</gml:limits>
<gml:axisName>u</gml:axisName>
<gml:axisName>v</gml:axisName>
<gml:origin>
  <gml:Point gml:id="palindrome" srsName="urn:EPSG:geographicCRS:4327">
    <gml:pos>1.2 3.3 2.1</gml:pos>
  </gml:Point>
</gml:origin>
<gml:offsetVector srsName="urn:EPSG:geographicCRS:4327">1.1 2.2 3.3</gml:offsetVector>
<gml:offsetVector srsName="urn:EPSG:geographicCRS:4327">2.0 1.0 0.0</gml:offsetVector>
</gml:RectifiedGrid>

```

Note that in this example the RectifiedGrid starts at integer offset 1 1 (value of `low` property) relative to the origin as shown in Figure 6.



**Figure 6 – RectifiedGrid with non-zero low limit**

## 19.3 Coverage schema

The coverage.xsd schema is listed in Annex C. It is identified by the following location-independent name (using URN syntax):

urn:opengis:specification:gml:schema-xsd:coverage:v3.1.0

### 19.3.1 gml:AbstractCoverageType, gml:\_Coverage

The base type for coverages is AbstractCoverageType, defined in the schema as follows:

```

<complexType name="AbstractCoverageType" abstract="true">
  <complexContent>
    <extension base="gml:BoundedFeatureType">
      <sequence>
        <element ref="gml:domainSet"/>
        <element ref="gml:rangeSet"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

NOTE gml:AbstractCoverageType is derived by extension from BoundedFeatureType, so any coverage whose content model is derived from this type is a GML feature, and the boundedBy property is mandatory on all coverages.

The basic elements of a coverage can be seen in this content model: the coverage contains domainSet and rangeSet properties. The gml:domainSet property describes the domain of the coverage and the rangeSet property describes the range of the coverage.

The abstract element `gml:_Coverage` is declared as follows:

```
<element name="_Coverage" type="gml:AbstractCoverageType" abstract="true" substitutionGroup="gml:_Feature"/>
```

This element serves as the head of a substitution group which may contain any coverage whose type is derived from `gml:AbstractCoverageType`. It may act as a variable in the definition of content models where it is required to permit any coverage to be valid.

### 19.3.2 `gml:AbstractDiscreteCoverageType`, `gml:_DiscreteCoverage`

A discrete coverage consists of a domain set, range set and optionally a coverage function. The domain set consists of either geometry or temporal objects, finite in number. The range set is comprised of a finite number of attribute values each of which is associated to every direct position within any single spatiotemporal object in the domain. In other words, the range values are constant on each spatiotemporal object in the domain. This coverage function maps each element from the coverage domain to an element in its range. This definition conforms to ISO 19123. The base type for discrete coverages is `AbstractDiscreteCoverageType`, defined in the schema as follows:

```
<complexType name="AbstractDiscreteCoverageType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractCoverageType">
      <sequence>
        <element ref="gml:coverageFunction" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

The `coverageFunction` element describes the mapping “ $f$ ” as shown in Figure 4.

The abstract element `gml:_DiscreteCoverage` is declared as follows:

```
<element name="_DiscreteCoverage" type="gml:AbstractDiscreteCoverageType" abstract="true"
substitutionGroup="gml:_Coverage"/>
```

This element serves as the head of a substitution group which may contain any discrete coverage whose type is derived from `gml:AbstractDiscreteCoverageType`.

### 19.3.3 `gml:AbstractContinuousCoverageType`, `gml:_ContinuousCoverage`

A continuous coverage as defined in ISO 19123 is a coverage that can return different values for the same feature attribute at different direct positions within a single spatiotemporal object in its spatiotemporal domain. This definition conforms to ISO 19123. The base type for continuous coverages is `AbstractContinuousCoverageType`, defined in the schema as follows:

```
<complexType name="AbstractContinuousCoverageType" abstract="true">
  <annotation>
    <documentation>A continuous coverage as defined in ISO 19123 is a coverage that can return different values for the same feature attribute at different direct positions within a single spatiotemporal object in its spatiotemporal domain</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoverageType">
      <sequence>
        <element ref="gml:coverageFunction" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

The coverageFunction element describes the mapping “f” as shown in Figure 4.

The abstract element gml:\_ContinuousCoverage is declared as follows:

```
<element name="_ContinuousCoverage" type="gml:AbstractContinuousCoverageType" abstract="true"
substitutionGroup="gml:_Feature"/>
```

This element serves as the head of a substitution group which may contain any continuous coverage whose type is derived from gml:AbstractContinuousCoverageType.

#### 19.3.4 gml:domainSet, gml:DomainSetType

The Domain Set describes the spatio-temporal region of interest, within which the coverage is defined. Its content model is given by gml:DomainSetType which is defined as follows:

```
<element name="domainSet" type="gml:DomainSetType"/>

<complexType name="DomainSetType">
  <choice minOccurs="0">
    <element ref="gml:_Geometry"/>
    <element ref="gml:_TimeObject"/>
  </choice>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

The value of the domain is thus a choice between a gml:\_Geometry and a gml:\_TimeObject. In the instance these abstract elements will normally be substituted by a geometry complex or temporal complex, to represent spatial coverages and time-series, respectively.

**NOTE** Spatiotemporal domains are supported if the domain is described using a compound coordinate reference system, one of whose components is temporal. Otherwise, following the ISO 19100 and OGC abstract specifications, GML 3 does not support combined spatial-temporal domains.

The presence of the gml:AssociationAttributeGroup means that domainSet follows the usual GML property model and can use the xlink:href attribute to point to the Domain, as an alternative to describing the domain inline.

#### 19.3.5 gml:rangeSet, gml:RangeSetType

The Range Set contains the values of the coverage (sometimes called the attribute values). Its content model is given by gml:RangeSetType which is defined as follows::

```
<element name="rangeSet" type="gml:RangeSetType"/>

<complexType name="RangeSetType">
  <choice>
    <element ref="gml:ValueArray" maxOccurs="unbounded"/>
    <element ref="gml:_ScalarValueList" maxOccurs="unbounded"/>
    <element ref="gml:_DataBlock"/>
    <element ref="gml:File"/>
  </choice>
</complexType>
```

This content model supports a structural description of the Range. The semantic information describing the range set is embedded using a uniform method, as part of the explicit values, or as a template value accompanying the representation using gml:\_DataBlock and gml:File.

- c) The values from each component (or “band”) in the range may be encoded within a gml:ValueArray element or a concrete member of the gml:\_ScalarValueList substitution group (e.g. gml:CategoryList, gml:QuantityList - see clause 16.4). Use of these elements satisfies the value-type homogeneity requirement

### 19.3.6 gml:DataBlock

gml:DataBlock describes the Range as a block of text encoded values similar to a Common Separated Value (CSV) representation. The content model is as follows:

```
<element name="DataBlock" type="gml:DataBlockType"/>

<complexType name="DataBlockType">
  <sequence>
    <element ref="gml:rangeParameters"/>
    <choice>
      <element ref="gml:tupleList"/>
      <element ref="gml:doubleOrNullTupleList"/>
    </choice>
  </sequence>
</complexType>
```

The range set parameterization is described by the property gml:rangeParameters.

### 19.3.7 gml:rangeParameters

The gml:rangeParameters property is declared as follows:

```
<element name="rangeParameters" type="gml:RangeParametersType"/>

<complexType name="RangeParametersType">
  <sequence>
    <element ref="gml:_Value" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

gml:RangeParameterType is a framework for the description of the range parameters each of which is a gml:\_Value, as described in clause 16.4. Specific range parameters are defined through the creation of a GML Application Schema that provides elements that are substitutable for gml:\_Value.

### 19.3.8 gml:tupleList

The gml:tupleList property is declared as follows:

```
<element name="tupleList" type="gml:CoordinatesType"/>
```

gml:CoordinatesType is described in clause 9.1.3.3. It consists of a list of coordinate tuples, with each coordinate tuple separated by the ts or tuple separator (by default this is whitespace), and each coordinate in the tuple by the cs or coordinate separator (by default this is a comma).

The gml:tupleList encoding is effectively “band-interleaved”. An example of a set of pairs of temperature and pressure observations might be recorded in a gml:DataBlock as follows:

```
<gml:DataBlock>
  <gml:rangeParameters>
    <gml:CompositeValue>
      <gml:valueComponents>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">template</Temperature>
        <Pressure uom="urn:x-si:v1999:uom:kPa">template</Pressure>
      </gml:valueComponents>
    </gml:CompositeValue>
  </gml:rangeParameters>
  <gml:tupleList>3,101.2 5,101.3 7,101.4 11,101.5 13,101.6 17,101.7 19,101.7 23,101.8 29,101.9 31,102.0
37,102.1 41,102.2 43,102.3 47,102.4 53,102.5 59,102.6</gml:tupleList>
</gml:DataBlock>
```

where Temperature and Pressure are elements defined in a local application schema, using gml:MeasureOrNullListType.

### 19.3.9 gml:doubleOrNullTupleList

The gml:doubleOrNullTupleList property is declared as follows:

```
<element name="doubleOrNullTupleList" type="gml:doubleOrNullList"/>
```

gml:doubleOrNullList is described in clause 7.3.4.1. It consists of a list of doubleOrNull values, each separated by a whitespace. The doubleOrNull values are grouped into tuples where the dimension of each tuple in the list is equal to the number of range parameters.

An example of the use of gml:doubleOrNullTupleList to record the same set of pairs of temperature and pressure observations given in the gml:DataBlock example above is as follows:

```
<gml:DataBlock>
  <gml:rangeParameters>
    <gml:CompositeValue>
      <gml:valueComponents>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">template</Temperature>
        <Pressure uom="urn:x-si:v1999:uom:kPa">template</Pressure>
      </gml:valueComponents>
    </gml:CompositeValue>
  </gml:rangeParameters>
  <gml:doubleOrNullTupleList>3 101.2 5 101.3 7 101.4 11 101.5 13 101.6 17 101.7 19 101.7 23 101.8 29
101.9 31 102.0 37 102.1 41 102.2 43 102.3 47 102.4 53 102.5 59 102.6</gml:doubleOrNullTupleList>
</gml:DataBlock>
```

### 19.3.10 gml:File, gml:FileType

For efficiency reasons, GML 3 also provides a means of encoding the range set in an arbitrary external encoding, such as a binary file. This encoding may be “well-known” but this is not required. This mode uses the gml:File element, which is declared as follows:

```
<element name="File" type="gml:FileType"/>

<complexType name="FileType">
  <sequence>
    <element ref="gml:rangeParameters"/>
    <element name="fileName" type="anyURI"/>
    <element name="fileStructure" type="gml:FileValueModelType"/>
    <element name="mimeType" type="anyURI" minOccurs="0"/>
    <element name="compression" type="anyURI" minOccurs="0"/>
  </sequence>
</complexType>
```

In this version of the coverage encoding, the values of the coverage (attribute values in the Range set) are transmitted in a binary file that is referenced from the XML structure described by gml:FileType. The binary file is referenced by the fileName property that is an anyURI. This means that the binary file can be located remotely from the referencing GML instance. This can support, for example, both an http reference and a SOAP attachment.

The compression property points to a definition of a compression algorithm through an anyURI. This may be a retrievable, computable definition or simply a reference to an unambiguous name for the compression method.

The mime type property points to a definition of the file mime type.

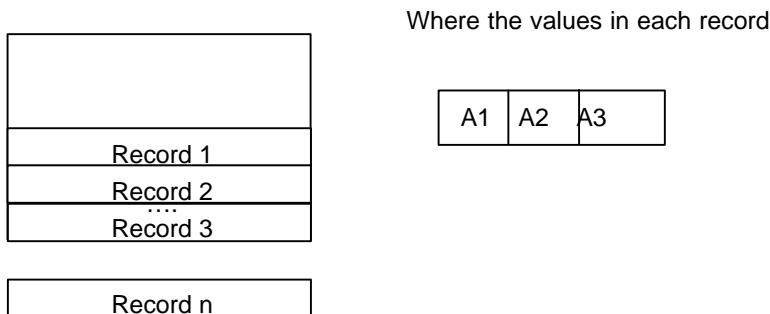
The fileStructure property is defined by the `gml:FileValueModelType`. This is simple enumerated type restriction on string. The only value supported in GML 3 is “Record Interleaved”. Additional values may be supported in future releases of GML. Note further that all values shall be enclosed in a single file. Multi-file structures for values are not supported in GML 3.

The semantics of the Range set is described as above using `gml:rangeParameters`.

EXAMPLE      Binary File Encoding:

```
<gml:File>
  <gml:rangeParameters>
    <gml:CompositeValue>
      <gml:valueComponents>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">template</Temperature>
        <Pressure uom="urn:x-si:v1999:uom:kPa">template</Pressure>
      </gml:valueComponents>
    </gml:CompositeValue>
  </gml:rangeParameters>
  <gml:fileName>http://www.somedata.org/temp_pressure.dat</gml:fileName>
  <gml:fileStructure>Record Interleaved</gml:fileStructure>
</gml:File>
```

The referenced file structure shall be as follows:



**Figure 7 – File Record Structure or Coverage File**

Note that if any compression algorithm is applied, the structure above applies only to the pre-compression or post-decompression structure of the file.

Note that the fields within a record match the valueComponents of the CompositeValue in document order.

#### 19.3.11 **gml:coverageFunction, gml:CovarianceFunctionType**

This clause describes the Coverage Function (gml:coverageFunction) property, that is, the mapping “f” (see Figure 4) from the domain to the range of the coverage. The content model for the coverage function is given by:

```
<element name="coverageFunction" type="gml:CovarianceFunctionType"/>

<complexType name="CovarianceFunctionType">
  <choice>
    <element ref="gml:MappingRule"/>
    <element ref="gml:GridFunction"/>
  </choice>
</complexType>
```

Note that the value of the CoverageFunction is one of gml:MappingRule and gml:GridFunction.

#### 19.3.12 **gml:MappingRule**

gml:MappingRule provides a formal (e.g. MathML) or informal (e.g. free text) description of the coverage function, per:

```
<element name="MappingRule" type="gml:StringOrRefType"/>
```

The mapping rule may be defined as an in-line string or via a remote reference through xlink:href.

If no rule name is specified, the default is ‘Linear’ with respect to members of the domain in document order.

#### 19.3.13 **gml:GridFunction, gml:GridFunctionType**

gml:GridFunction provides an explicit mapping rule for grid geometries, i.e. the domain shall be a geometry of type grid. It describes the mapping of grid posts (discrete point grid coverage) or grid cells (discrete surface coverage) to the values in the RangeSet. The content model is as follows:

```
<element name="GridFunction" type="gml:GridFunctionType"/>
```

```

<complexType name="GridFunctionType">
  <sequence>
    <element name="sequenceRule" type="gml:SequenceRuleType" minOccurs="0"/>
    <element name="startPoint" type="gml:integerList" minOccurs="0">
    </sequence>
</complexType>

```

The startPoint is the index position of a point in the grid that is mapped to the first point in the Range Set (this is also the index position of the first grid post). *If the startPoint is omitted the startPoint is assumed to be equal to the value of gml:low in the gml:Grid geometry.* Subsequent points in the mapping are determined by the value of the sequenceRule.

#### 19.3.14 gml:sequenceRule, gml:SequenceRuleType, gml:SequenceRuleNames

The sequenceRule is described by the content model:

```

<complexType name="SequenceRuleType">
  <simpleContent>
    <extension base="gml:SequenceRuleNames">
      <attribute name="order" type="gml:IncrementOrder" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

```

The SequenceRuleType is derived from the gml:SequenceRuleNames through the addition of an order attribute. The gml:SequenceRuleNames is an enumerated type defined as:

```

<simpleType name="SequenceRuleNames">
  <restriction base="string">
    <enumeration value="Linear"/>
    <enumeration value="Boustrophedonic"/>
    <enumeration value="Cantor-diagonal"/>
    <enumeration value="Spiral"/>
    <enumeration value="Morton"/>
    <enumeration value="Hilbert"/>
  </restriction>
</simpleType>

```

These rule names are defined in ISO 19123.

*If no rule name is specified the default is "Linear".*

The order attribute is also defined in ISO 19123 and its value is determined by the content model:

```

<simpleType name="IncrementOrder">
  <annotation>
    <documentation></documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="+x+y"/>
    <enumeration value="+y+x"/>
    <enumeration value="+x-y"/>
    <enumeration value="-x-y"/>
  </restriction>
</simpleType>

```

The enumeration value here indicates the incrementing order to be used on the first 2 axes, i.e. "+x-y" means that the points on the first axis are to be traversed from lowest to highest and the points on the second axis are to be traversed from highest to lowest. The points on all other axes (if any) beyond the first 2 are assumed to increment from lowest to highest.

If the order attribute is omitted it is assumed to have the value “+x+y”.

If the coverageFunction property is omitted for Gridded Coverages (included RectifiedGridded Coverages) the startPoint is assumed to be the value of the gml:low property in the gml:Grid geometry, and the sequenceRule is assumed to be linear and the order attribute is assumed to be “+x+y”. This is best illustrated by a simple example as follows:

```

<AverageTempPressure
  xmlns="http://www.opengis.net/app" xmlns:gml="http://www.opengis.net/gml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/app ./CoverageExamples.xsd">
  <gml:domainSet>
    <gml:Grid dimension="2">
      <gml:limits>
        <gml:GridEnvelope>
          <gml:low>0 0</gml:low>
          <gml:high>4 4</gml:high>
        </gml:GridEnvelope>
      </gml:limits>
      <gml:axisName>x</gml:axisName>
      <gml:axisName>y</gml:axisName>
    </gml:Grid>
  </gml:domainSet>
  <gml:rangeSet>
    <gml:DataBlock>
      <gml:rangeParameters>
        <gml:CompositeValue>
          <gml:valueComponents>
            <Temperature uom="urn:x-si:v1999:uom:degreesC">template</Temperature>
            <Pressure uom="urn:x-si:v1999:uom:kPa">template</Pressure>
          </gml:valueComponents>
        </gml:CompositeValue>
      </gml:rangeParameters>
      <gml:tupleList>3,101.2 5,101.3 7,101.4 11,101.5 13,101.6 17,101.7 19,101.7 23,101.8 29,101.9 31,102.0
37,102.1 41,102.2 43,102.3 47,102.4 53,102.5 59,102.6</gml:tupleList>
    </gml:DataBlock>
  </gml:rangeSet>
</AverageTempPressure>
```

Since no coverageFunction is specified the function is assumed to be that of linear scanning with “+x+y” order starting at the location (0 0). If we look at the DataBlock we see that we have the following mapping.

**Table 5 – Data block example**

Grid Location	Data Value
0 0	3,101.2
1 0	5,101.3
2 0	7,101.4
3 0	11,101.5
0 1	13,101.6
1 1	17,101.7
2 1	19,101.7
3 1	23,101.8
0 2	29,101.9
1 2	31,102.0
2 2	37,102.1
3 2	41,102.2

0 3	43,102.3
1 3	47,102.4
2 3	53,102.5
3 3	59,102.6

### 19.3.15 Specific Coverage Types in GML 3.1

GML 3 supports all of the discrete coverage types defined in ISO 19123.

The supported types are derived from `gml:AbstractDiscreteCoverageType` and include:

- `MultiPointCoverage`
- `MultiCurveCoverage`
- `MultiSurfaceCoverage`
- `MultiSolidCoverage`
- Gridded Coverage (discrete point coverage)
- Rectified Grid Coverage (discrete point coverage)

Additional specific continuous coverage types can be anticipated in future releases of GML. Users can also construct their own coverage types by deriving from `gml:AbstractDiscreteCoverageType`, `AbstractContinuousCoverageType` or by derivation from the specific concrete coverage types above.

Note that the same Range Set encodings apply for each of the different discrete coverage types as the latter are specified by the geometry type of the domain.

### 19.3.16 MultiPoint Coverage

In a MultiPoint Coverage the domain set is a `gml:MultiPoint`, that is a collection of arbitrarily distributed geometric points. The content model for a MultiPoint Coverage is as follows:

```
<complexType name="MultiPointCoverageType">
  <complexContent>
    <restriction base="gml:AbstractCoverageType">
      <sequence>
        <group ref="gml:StandardObjectProperties"/>
        <element ref="gml:boundedBy"/>
        <element ref="gml:multiPointDomain"/>
        <element ref="gml:rangeSet"/>
        <element ref="gml:coverageFunction" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
```

Note that this is defined by restriction on `gml:AbstractCoverageType`. Note that the restriction replaces the generic `gml:domainSet` by the specific `gml:multiPointDomain` whose value is a `gml:MultiPoint`.

```
<complexType name="MultiPointDomainType">
  <complexContent>
    <restriction base="gml:DomainSetType">
      <sequence minOccurs="0">
```

```

<element ref="gml:MultiPoint"/>
</sequence>
</restriction>
</complexContent>
</complexType>

```

Note that in a MultiPoint Coverage the mapping from the domain to the range is straightforward.

- For DataBlock encodings the points of the MultiPoint are mapped in document order to the tuples of the DataBlock.
- For CompositeValue encodings the points of the MultiPoint are mapped to the members of the CompositeValue in document order.
- For File encodings the points of the MultiPoint are mapped to the records of the File in sequential order.

EXAMPLE      MultiPoint Coverage (uses Value encoding):

```

<AverageTempPressure
  xmlns="http://www.opengis.net/app"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/app CoverageExamples.xsd">
  <gml:domainSet>
    <gml:MultiPoint srsName="urn:EPSG:geographicCRS:4326">
      <gml:pointMember>
        <gml:Point>
          <gml:pos>1 1</gml:pos>
        </gml:Point>
      </gml:pointMember>
      <gml:pointMember>
        <gml:Point>
          <gml:pos>2 2</gml:pos>
        </gml:Point>
      </gml:pointMember>
      <gml:pointMember>
        <gml:Point>
          <gml:pos>3 3</gml:pos>
        </gml:Point>
      </gml:pointMember>
      <gml:pointMember>
        <gml:Point>
          <gml:pos>4 4</gml:pos>
        </gml:Point>
      </gml:pointMember>
    </gml:MultiPoint>
  </gml:domainSet>
  <gml:rangeSet>
    <gml:ValueArray>
      <gml:valueComponents>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">3</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">5</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">7</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">11</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">13</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">17</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">19</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">23</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">29</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">31</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">37</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">41</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">43</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">47</Temperature>
      </gml:valueComponents>
    </gml:ValueArray>
  </gml:rangeSet>
</AverageTempPressure>

```

```

        <Temperature uom="urn:x-si:v1999:uom:degreesC">53</Temperature>
        <Temperature uom="urn:x-si:v1999:uom:degreesC">59</Temperature>
    </gml:valueComponents>
</gml:ValueArray>
</gml:rangeSet>
</AverageTempPressure>

```

### 19.3.17 MultiCurve Coverage

In a multi-curve coverage the domain is partitioned into a collection of curve elements comprising a gml:MultiCurve. The coverage function than maps each curve element in the collection to a value in the Range Set. The content model for the MultiCurve coverage is as follows:

```

<complexType name="MultiCurveCoverageType">
<complexContent>
<restriction base="gml:AbstractCoverageType">
<sequence>
<group ref="gml:StandardObjectProperties"/>
<element ref="gml:boundedBy"/>
<element ref="gml:multiCurveDomain"/>
<element ref="gml:rangeSet"/>
<element ref="gml:coverageFunction" minOccurs="0"/>
</sequence>
</restriction>
</complexContent>
</complexType>

```

Note that the domainSet is now a multiCurveDomain with value a gml:MultiCurve.

The coverage function provides a mapping from the elements of the MultiCurve to the values in the Range Set.

In the DataBlock encoding, the members of the MultiCurve (value of the multiCurveDomain) are mapped to the tuples in the DataBlock in document order.

In the File encoding, the members of the MultiCurve are mapped to the records in the File in document order.

In the CompositeValue encoding the members of the MultiCurve are mapped to the members of the CompositeValue in document order.

EXAMPLE      MultiCurve Coverage (uses DataBlock encoding):

```

<AverageTempPressure
  xmlns="http://www.opengis.net/app"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/app CoverageExamples.xsd">
<gml:domainSet>
  <gml:MultiCurve srsName="urn:EPSG:geographicCRS:4326">
    <gml:curveMember>
      <gml:LineString>
        <gml:posList dimension="2">1.1 1.1 2.2 2.2</gml:posList>
      </gml:LineString>
    </gml:curveMember>
    <gml:curveMember>
      <gml:LineString>
        <gml:posList dimension="2">2.2 2.2 3.3 3.3</gml:posList>
      </gml:LineString>
    </gml:curveMember>
    <gml:curveMember>
      <gml:LineString>

```

```

<gml:posList dimension="2">3.3 3.3 4.4 4.4</gml:posList>
</gml:LineString>
</gml:curveMember>
<gml:curveMember>
<gml:LineString>
<gml:posList dimension="2">4.4 4.4 5.5 5.5</gml:posList>
</gml:LineString>
</gml:curveMember>
</gml:MultiCurve>
</gml:domainSet>
<gml:rangeSet>
<gml:DataBlock>
<gml:rangeParameters>
<gml:CompositeValue>
<gml:valueComponents>
<Temperature uom="urn:x-si:v1999:uom:degreesC">template</Temperature>
<Pressure uom="urn:x-si:v1999:uom:kPa">template</Pressure>
</gml:valueComponents>
</gml:CompositeValue>
</gml:rangeParameters>
<gml:doubleOrNullTupleList>3 101.2 5 101.3 7 101.4 11 101.5</gml:doubleOrNullTupleList>
</gml:DataBlock>
</gml:rangeSet>
</AverageTempPressure>

```

### 19.3.18 MultiSurface Coverage

In a multi-surface coverage the domain is partitioned into a collection of surface elements comprising a gml:MultiSurface. The coverage function than maps each surface element in the collection to a value in the Range Set. The content model for the MultiSurface coverage is as follows:

```

<complexType name="MultiSurfaceCoverageType">
<complexContent>
<restriction base="gml:AbstractCoverageType">
<sequence>
<group ref="gml:StandardObjectProperties"/>
<element ref="gml:boundedBy"/>
<element ref="gml:multiSurfaceDomain"/>
<element ref="gml:rangeSet"/>
<element ref="gml:coverageFunction" minOccurs="0"/>
</sequence>
</restriction>
</complexContent>
</complexType>

```

Note that the domainSet is now a multiSurfaceDomain with value a gml:MultiSurface.

The coverage function provides a mapping from the elements of the MultiSurface to the values in the Range Set.

In the DataBlock encoding, the members of the MultiSurface (value of the multiSurfaceDomain) are mapped to the tuples in the DataBlock in document order.

In the File encoding, the members of the MultiSurface are mapped to the records in the File in document order.

In the CompositeValue encoding the members of the MultiSurface are mapped to the members of the CompositeValue in document order.

EXAMPLE      MultiSurface Coverage (uses File encoding for values):

```

<SoilData xmlns="http://www.opengis.net/app" xmlns:gml="http://www.opengis.net/gml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.opengis.net/app
./CoverageExamples.xsd">
  <gml:domainSet>
    <gml:MultiSurface srsName="urn:EPSG:geographicCRS:4326">
      <gml:surfaceMember>
        <gml:Polygon gid="p1">
          <gml:exterior>
            <gml:LinearRing>
              <gml:posList dimension="3">1. 2. 3.</gml:posList>
            </gml:LinearRing>
          </gml:exterior>
        </gml:Polygon>
      </gml:surfaceMember>
      <gml:surfaceMember>
        <gml:Polygon gid="p6">
          <gml:exterior>
            <gml:LinearRing>
              <gml:posList dimension="3">4. 5. 6.</gml:posList>
            </gml:LinearRing>
          </gml:exterior>
        </gml:Polygon>
      </gml:surfaceMember>
      <gml:surfaceMember>
        <gml:Polygon gid="p11">
          <gml:exterior>
            <gml:LinearRing>
              <gml:posList dimension="3">7. 8. 9.</gml:posList>
            </gml:LinearRing>
          </gml:exterior>
        </gml:Polygon>
      </gml:surfaceMember>
      <gml:surfaceMember>
        <gml:Polygon gid="p16">
          <gml:exterior>
            <gml:LinearRing>
              <gml:posList dimension="3">10. 11. 12.</gml:posList>
            </gml:LinearRing>
          </gml:exterior>
        </gml:Polygon>
      </gml:surfaceMember>
    </gml:MultiSurface>
  </gml:domainSet>
  <gml:rangeSet>
    <gml:File>
      <gml:rangeParameters>
        <gml:CompositeValue>
          <gml:valueComponent>
            <SoilType codeSpace="http://my.big.org/classifications/soils">template</SoilType>
          </gml:valueComponent>
          <gml:valueComponent>
            <SoilMoisture uom="http://my.big.org/units/percent">template</SoilMoisture>
          </gml:valueComponent>
        </gml:CompositeValue>
      </gml:rangeParameters>
      <gml:fileName>soil.dat</gml:fileName>
      <gml:fileStructure>Record Interleaved</gml:fileStructure>
    </gml:File>
  </gml:rangeSet>

```

### 19.3.19 MultiSolid Coverage

In a multi-solid coverage the domain is partitioned into a collection of solid elements comprising a gml:MultiSolid. The coverage function than maps each solid element in the collection to a value in the Range Set. The content model for the MultiSolid coverage is as follows:

```
<complexType name="MultiSolidCoverageType">
<complexContent>
<restriction base="gml:AbstractCoverageType">
<sequence>
<group ref="gml:StandardObjectProperties"/>
<element ref="gml:boundedBy"/>
<element ref="gml:multiSolidDomain"/>
<element ref="gml:rangeSet"/>
<element ref="gml:coverageFunction" minOccurs="0"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

Note that the domainSet is now a multiSolidDomain with value a gml:MultiSolid.

The coverage function provides a mapping from the elements of the MultiSolid to the values in the Range Set.

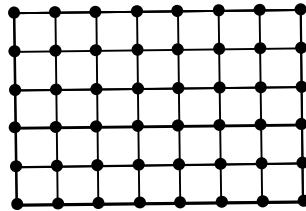
In the DataBlock encoding, the members of the MultiSolid (value of the multiSolidDomain) are mapped to the tuples in the DataBlock in document order.

In the File encoding, the members of the MultiSolid are mapped to the records in the File in document order.

In the CompositeValue encoding the members of the MultiSolid are mapped to the members of the CompositeValue in document order.

### 19.3.20 Gridded Coverage

A gridded coverage is a discrete point coverage in which the domain set is a geometric grid of points as shown in Figure 8.



**Figure 8 – Gridded Coverage domain is a grid of points**

The content model for the gridded coverage is as follows:

```
<element name="GridCoverage" type="gml:GridCoverageType" substitutionGroup="gml:_Coverage"/>

<complexType name="GridCoverageType">
<complexContent>
<restriction base="gml:AbstractCoverageType">
<sequence>
<group ref="gml:StandardObjectProperties"/>
<element ref="gml:boundedBy"/>
<element ref="gml:gridDomain"/>
<element ref="gml:rangeSet"/>
<element ref="gml:coverageFunction" minOccurs="0"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

```

    </restriction>
</complexContent>
</complexType>
```

Note that this is the same as the MultiPoint coverage except that we have a gml:gridDomain property to describe the domain.

The gml:gridDomain is defined as:

```

<complexType name="GridDomainType">
<complexContent>
<restriction base="gml:DomainSetType">
<choice minOccurs="0">
<element ref="gml:Grid"/>
</choice>
</restriction>
</complexContent>
</complexType>
```

The gml:Grid is defined in the grids.xsd discussed in Clause 19.2. Note that the simple gridded coverage is not geometrically referenced and hence no geometric positions are assignable to the points in the grid. Such geometric positioning is introduced in the RectifiedGrid Coverage discussed in Clause 19.2.2.

**EXAMPLE** Grid Coverage (uses File encoding for values):

```

<AverageTempPressure>
<gml:gridDomain>
<gml:Grid dimension="2">
<gml:limits>
<gml:GridEnvelope>
<gml:low>0 0</gml:low>
<gml:high>4 4</gml:high>
</gml:GridEnvelope>
</gml:limits>
<gml:axisName>x</gml:axisName>
<gml:axisName>y</gml:axisName>
</gml:Grid>
</gml:gridDomain>
<gml:rangeSet>
<gml:File>
<gml:rangeParameters>
<gml:CompositeValue>
<gml:valueComponents>
<Temperature uom="urn:x-si:v1999:uom:degreesC">template</Temperature>
<Pressure uom="urn:x-si:v1999:uom:kPa">template</Pressure>
</gml:valueComponents>
</gml:CompositeValue>
</gml:rangeParameters>
<gml:fileName>http://www.somedata.org/temp\_pressure.dat</gml:fileName>
<gml:fileStructure>Record Interleaved</gml:fileStructure>
</gml:File>
</gml:rangeSet>
</AverageTempPressure>
```

### 19.3.21 RectifiedGrid Coverage

The rectified grid coverage is a discrete point coverage based on a rectified grid.

The rectified grid coverage is similar to the grid coverage of Clause 19.2.2 except that the points of the grid are geometrically referenced. The rectified grid coverage has a domain that is a RectifiedGrid geometry as defined in the grids.xsd of Clause 19.2.

The content model for gml:RectifiedGridCoverage is as follows:

```
<complexType name="RectifiedGridCoverageType">
<complexContent>
<restriction base="gml:AbstractCoverageType">
<sequence>
<group ref="gml:StandardObjectProperties"/>
<element ref="gml:boundedBy"/>
<element ref="gml:rectifiedGridDomain"/>
<element ref="gml:rangeSet"/>
<element ref="gml:coverageFunction" minOccurs="0"/>
</sequence>
</restriction>
</complexContent>
</complexType>
```

Note that the coverage domain is described by gml:rectifiedGridDomain.

```
<complexType name="RectifiedGridDomainType">
<complexContent>
<restriction base="gml:DomainSetType">
<choice minOccurs="0">
<element ref="gml:RectifiedGrid"/>
</choice>
</restriction>
</complexContent>
</complexType>
```

where the gml:RectifiedGrid geometry is defined in grids.xsd discussed in Clause 19.2.

EXAMPLE RectifiedGrid Coverage (using DataBlock):

```
<AveragePressure xmlns="http://www.opengis.net/app" xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.opengis.net/app
/CoverageExamples.xsd">
<gml:rectifiedGridDomain>
<gml:RectifiedGrid dimension="2">
<gml:limits>
<gml:GridEnvelope>
<gml:low>1 1</gml:low>
<gml:high>4 4</gml:high>
</gml:GridEnvelope>
</gml:limits>
<gml:axisName>u</gml:axisName>
<gml:axisName>v</gml:axisName>
<gml:origin>
<gml:Point gml:id="palindrome" srsName="urn:EPSG:geographicCRS:4327">
<gml:pos>1.2 3.3 2.1</gml:pos>
</gml:Point>
</gml:origin>
<gml:offsetVector srsName="urn:EPSG:geographicCRS:4327">1.1 2.2 3.3</gml:offsetVector>
<gml:offsetVector srsName="urn:EPSG:geographicCRS:4327">2.0 1.0 0.0</gml:offsetVector>
</gml:RectifiedGrid>
</gml:rectifiedGridDomain>
<gml:rangeSet>
<gml>DataBlock>
<gml:rangeParameters>
<gml:CompositeValue>
<gml:valueComponents>
<Pressure uom="urn:x-si:v1999:uom:kPa">template</Pressure>
</gml:valueComponents>
</gml:CompositeValue>
</gml:rangeParameters>
```

```
<gml:doubleOrNullTupleList>101.2 101.3 101.4 101.5 101.6 101.7 101.7 101.8 101.9 102.0 102.1 102.2 102.3  
102.4 102.5 102.6</gml:doubleOrNullTupleList>  
</gml:DataBlock>  
</gml:rangeSet>  
</AveragePressure>
```

## 20 GML schemas – default styling

### 20.1 General concepts

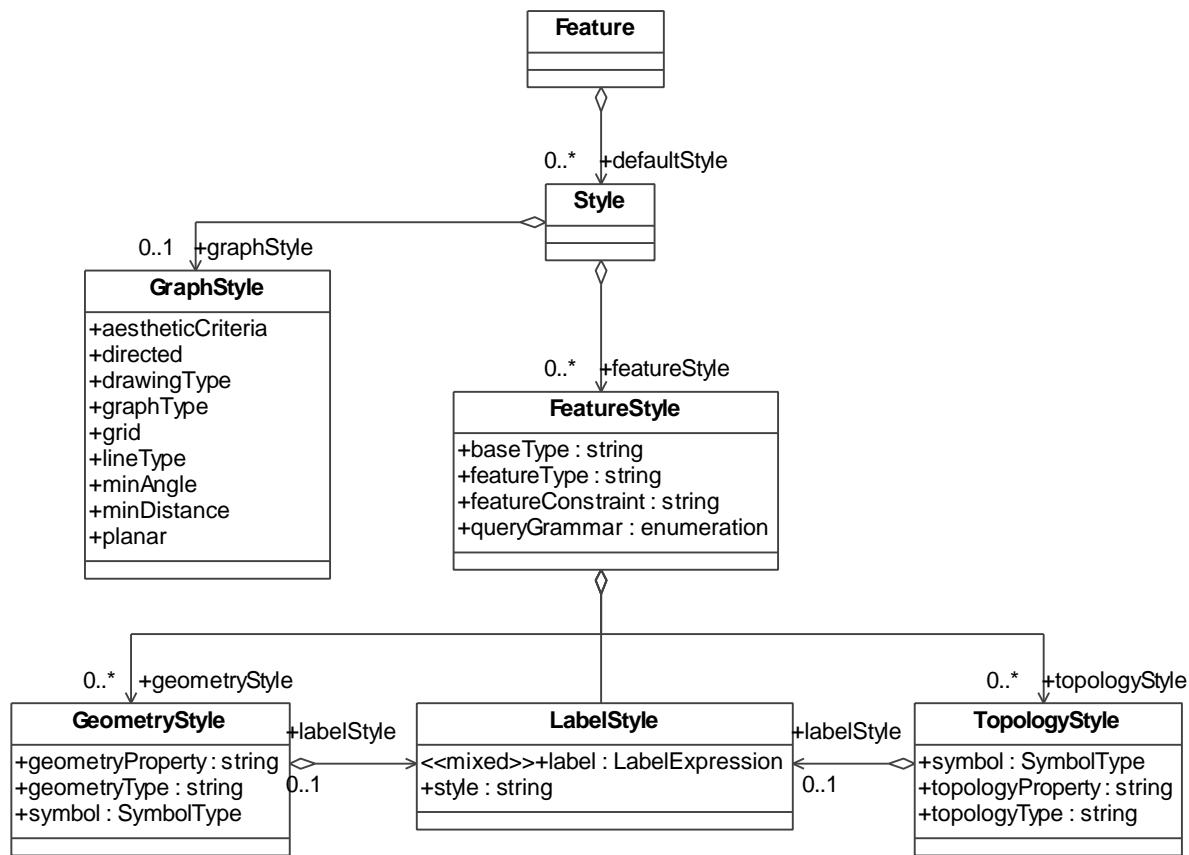
One of the requirements in developing of GML is the strict separation of data and presentation. Therefore none of the GML data description constructs have built-in capability to describe the styling information. Rather, the default styling mechanism was created as a separate model that can be “plugged-in” to a GML data set. An example of the use of such pluggable mechanism is data styling information persistence.

The term “default” signifies very relaxed relation to other parts of the GML model. The style information that is assigned to a data set may be used for styling but may also be completely ignored.

The default style schema is directly related to the rest of the GML schemas only in that it imports some very basic constructs from it. None of the other GML schemas depend on the styling one. It also depends on W3C Synchronized Multimedia Integration Language (SMIL) schemas.

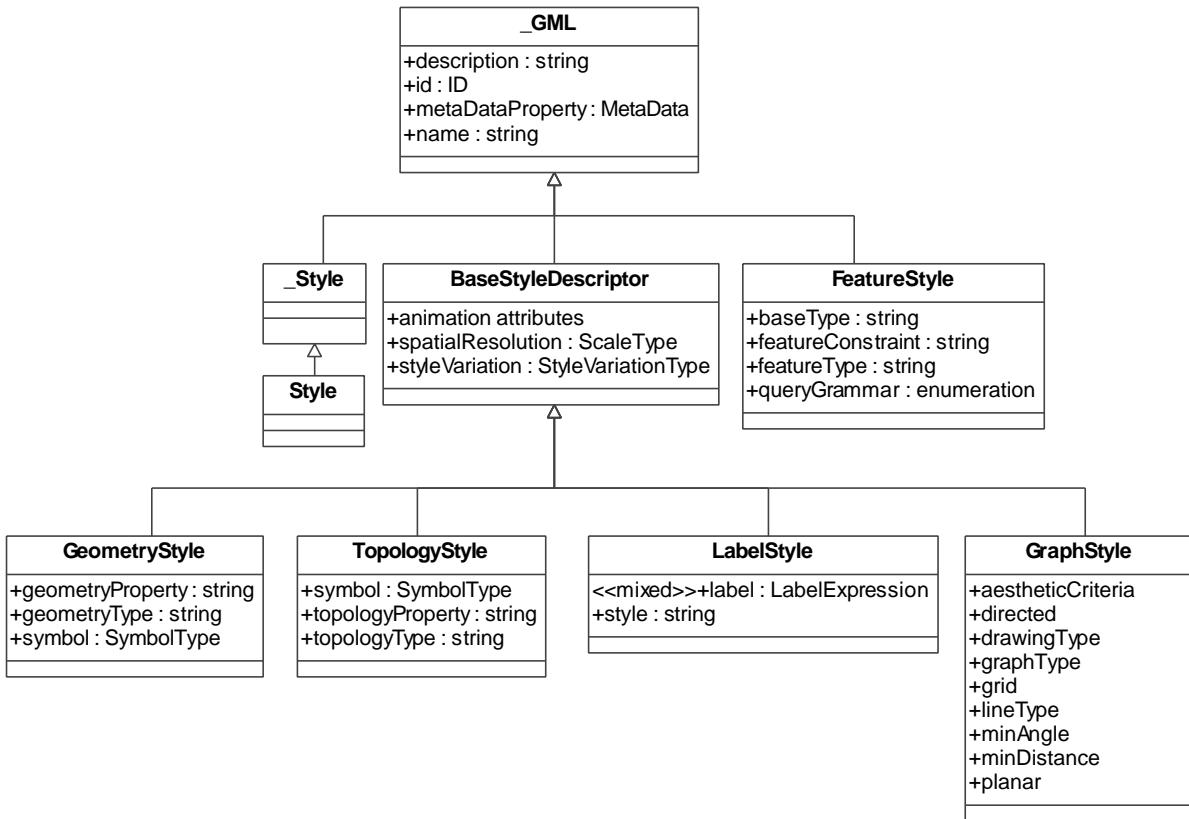
The relation of the default style information and GML data instances is achieved through the `gml:deafaultStyle` property defined in `defaultStyle.xsd` schema. The property may be assigned to the instance by defining such a relationship in application schema. Since GML is feature-based encoding, GML default style always applies to a feature, features or feature collections.

The UML containment diagram of the styling model is shown in Figure 9.



**Figure 9 – The GML default styling containment model diagram**

The UML inheritance diagram of the styling model is shown in Figure 10.



**Figure 10 – The GML default styling inheritance model diagram**

## 20.2 Top-level styling elements

The connection between a GML data set and a styling description is minimal. It is established through only one property, the `gml:defaultStyle`. The value of this property, the `gml:Style` object, contains all styling descriptions..

### 20.2.1 `gml:defaultStyle`

The `gml:defaultStyle` property is a property defined as a global element in GML namespace and thus can be assigned to any feature or a feature collection defined in an application schema. The definition of the property is shown in the following listing:

```

<element name="defaultStyle" type="gml:DefaultStylePropertyType"/>

<complexType name="DefaultStylePropertyType">
  <sequence>
    <element ref="gml:_Style" minOccurs="0"/>
  </sequence>
  <attribute name="about" type="anyURI" use="optional"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
  
```

To include this property in a feature users shall do so via the feature application schema. This is achieved by including it in a definition in an application schema. Following is an example of an exp:Road feature definition that includes the gml:defaultStyle property.

```
<element name="Road" type="exp:RoadType" substitutionGroup="gml:_Feature"/>

<complexType name="RoadType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:centerLineOf"/>
        <element ref="gml:defaultStyle"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

The gml:AssociationAttributeGroup on the property contains the simple XLink attribute group from xlink.xsd schema and the remoteSchema attribute. Attributes from xslinks.xsd schema allow a user to specify the value of the property remotely instead of inline. The remoteSchema attribute may be used to specify the schema of the property's value.

The about attribute on the property allows us to assign the style to an arbitrary feature or feature collection regardless of the style's location. The meaning of the attribute is that "The style is about (applies to) feature or features that can be found at the URL that is the attribute value".

## 20.2.2 gml:Style

The gml:Style object is the default concrete value of the gml:defaultStyle property. It is the top-level styling object that encapsulates all other, partial style descriptions. Its definition is presented in the following listing:

```
<element name="Style" type="gml:StyleType" substitutionGroup="gml:_Style"/>

<complexType name="StyleType">
  <complexContent>
    <extension base="gml:AbstractStyleType">
      <sequence>
        <element ref="gml:featureStyle" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:graphStyle" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

The gml:StyleType which is the content model of the gml:Style object extends the gml:AbstractStyleType. This base abstract type serves as an abstract base for extensibility purposes, i.e. creating custom style objects, and it does not add any new content to the gml:AbstractGMLType from which it derives. This can be seen in the listing below:

```
<element name="_Style" type="gml:AbstractStyleType" abstract="true" substitutionGroup="gml:_GML"/>

<complexType name="AbstractStyleType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractGMLType"/>
  </complexContent>
</complexType>
```

However, it is not assumed that creating custom style objects will be usual practice since gml:Style object provides rich capabilities for describing styles. In case of using this mechanism, usual GML rules have to be observed:

- d) The content model of the custom style object has to derive from gml:AbstractStyleType.

- e) The custom style object has to be substitutable for abstract `gml:_Style`.

The definition of the `gml:Style` object presented previously in the text is itself a proper example of using the extensibility mechanism and it shows how these rules are implemented in the schema.

The function of the styling elements in the `gml:Style` object, namely `gml:FeatureStyle` and `gml:GraphStyle` is to describe styles for two aspects of GML data: individual features and topology graphs that consist of collections of features. Note that elements that describe styles for particular aspects of features, namely, feature style, graph style, geometry style, topology style and label style are often called style descriptors.

## 20.3 Feature style

### 20.3.1 `gml:FeatureStyle`

Feature style descriptor is assigned to the `gml:Style` element using `gml:featureStyle` property. It allows, like other GML properties, to specify the value inline or remotely. For that purpose it has the attributes from the `gml:AssociationAttributeGroup` as well as the `about` attribute. Its schema is shown in the following listing:

```
<element name="featureStyle" type="gml:FeatureStylePropertyType"/>

<complexType name="FeatureStylePropertyType">
    <sequence>
        <element ref="gml:FeatureStyle" minOccurs="0"/>
    </sequence>
    <attribute name="about" type="anyURI" use="optional"/>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

Its value is `gml:FeatureStyle` – the feature style descriptor. Feature style descriptor describes the styling information for a set of feature instances. The set is defined by the selection mechanisms that are part of this style descriptor. The style applies to each feature in the set independently - no relations that might exist among features in the set are significant. (The opposite case is graph style where the style applies to a set of features as a whole). The definition of the feature style descriptor is shown in the following listing:

```
<element name="FeatureStyle" type="gml:FeatureStyleType" substitutionGroup="gml:_GML"/>

<complexType name="FeatureStyleType">
    <complexContent>
        <extension base="gml:AbstractGMLType">
            <sequence>
                <element name="featureConstraint" type="string" minOccurs="0"/>
                <element ref="gml:geometryStyle" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:topologyStyle" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:labelStyle" minOccurs="0"/>
            </sequence>
            <attribute name="featureType" type="string" use="optional"/>
            <attribute name="baseType" type="string" use="optional"/>
            <attribute name="queryGrammar" type="gml:QueryGrammarEnumeration"/>
        </extension>
    </complexContent>
</complexType>
```

Feature instances to which the style applies are selected using one of the attributes `featureType` or `baseType` and `gml:featureConstraint` element. These two attributes shall be used exclusively, with or without the `gml:featureConstraint` element.

### 20.3.2 `featureType`

The simplest and most common way of relating features and styles is by using this attribute. Its value will be the declared name of a feature, instances of which we want to style. For example, if the value is `exp:Road`, the

gml:FeatureStyle object will simply apply to all Road features. The value of this attribute is always the name of the element from the application schema that declares the feature.

### 20.3.3 **baseType**

Another way of selecting the feature instances to which the style applies is to specify, as the value of this attribute, the name of the base type from which feature or features derive. This is always the name of an XMLSchema complex type. Any complex type from the derivation chain can be used; the style applies to any feature instance that ultimately derives from it. If we use, for example, gml:AbstractFeatureType as the attribute's value, the style applies to all feature instances in a data set.

### 20.3.4 **featureConstraint**

This property is used to further constrain the feature instance set to which the style applies. It is optional and its value is an XPath expression. If the property does not exist, the style applies to all feature instances selected by featureType or baseType attribute.

### 20.3.5 **queryGrammar**

The value of this property which is defined as an enumeration specifies the grammar that is used in the content of the gml:featureConstraint element. The enumeration allows for three values: Xpath, Xquery and other.

Styling features means styling a particular aspect or aspects of a feature. We can style feature geometry, topology or display arbitrary text string. Feature style contains three style descriptors for respective purposes: gml:GeometryStyle, gml:TopologyStyle and gml:LabelStyle.

## 20.4 Geometry Style

The value of the gml:geometryStyle property is gml:GeometryStyle descriptor which describes the style for one geometry of a feature. Any number of geometry style descriptors can be assigned to one feature style. This is usually required for features with multiple geometry properties.

Geometry style property and descriptor are defined as follows:

```

<element name="geometryStyle" type="gml:GeometryStylePropertyType"/>

<complexType name="GeometryStylePropertyType">
  <sequence>
    <element ref="gml:GeometryStyle" minOccurs="0"/>
  </sequence>
  <attribute name="about" type="anyURI" use="optional"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="GeometryStyle" type="gml:GeometryStyleType" substitutionGroup="gml:_GML"/>

<complexType name="GeometryStyleType">
  <complexContent>
    <extension base="gml:BaseStyleDescriptorType">
      <sequence>
        <choice>
          <element ref="gml:symbol"/>
          <element name="style" type="string">
            <annotation>
              <appinfo>deprecated</appinfo>
              <documentation>
                Deprecated in GML version 3.1.0. Use symbol with inline content instead.
              </documentation>
            </annotation>
          </element>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

        </choice>
        <element ref="gml:labelStyle" minOccurs="0"/>
    </sequence>
    <attribute name="geometryProperty" type="string"/>
    <attribute name="geometryType" type="string"/>
</extension>
</complexContent>
</complexType>
```

The `gml:geometryStyle` is defined in the same manner as other GML properties which allow for referencing the value remotely or inline.

The `geometryProperty` attribute on the `gml:GeometryStyle` specifies the name of the geometry property of a feature to which this geometry style descriptor applies. It is necessary to specify the geometry type using `geometryType` attribute as well since the application schema of the geometry property may allow different geometries as its value.

Element `gml:symbol` is described in the section 20.7.2.

## 20.5 Topology Style

The value of the `gml:topologyStyle` property is `gml:TopologyStyle` descriptor which describes the style for one topology property. Similarly to the `gml:Geometry` style, a feature can have multiple topology properties, thus multiple topology style descriptors can be specified within one feature style.

The definition of topology style property and descriptor are presented in the following listing:

```

<element name="topologyStyle" type="gml:TopologyStylePropertyType"/>

<complexType name="TopologyStylePropertyType">
    <sequence>
        <element ref="gml:TopologyStyle" minOccurs="0"/>
    </sequence>
    <attribute name="about" type="anyURI" use="optional"/>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="TopologyStyle" type="gml:TopologyStyleType" substitutionGroup="gml:_GML"/>

<complexType name="TopologyStyleType">
    <complexContent>
        <extension base="gml:BaseStyleDescriptorType">
            <sequence>
                <choice>
                    <element ref="gml:symbol"/>
                    <element name="style" type="string">
                        <annotation>
                            <appinfo>deprecated</appinfo>
                            <documentation>
                                Deprecated in GML version 3.1.0. Use symbol with inline content instead.
                            </documentation>
                        </annotation>
                    </element>
                </choice>
                <element ref="gml:labelStyle" minOccurs="0"/>
            </sequence>
            <attribute name="topologyProperty" type="string"/>
            <attribute name="topologyType" type="string"/>
        </extension>
    </complexContent>
</complexType>
```

The `gml:topologyStyle` property is defined in the same manner as other GML properties which allow for referencing the value remotely or inline.

The `topologyProperty` attribute on the `gml:TopologyStyle` descriptor specifies the name of the topology property of a feature to which this topology style descriptor applies. It is necessary to specify the topology type using `topologyType` attribute as well since the application schema of the topology property may allow different topologies as its value.

Element `gml:symbol` is described in the section 20.7.2.

## 20.6 Label Style

The value of the `gml:labelStyle` property is `gml:LabelStyle` descriptor which describes the style for the text that is to be displayed along with the graphical representation of a feature. The content of the label is not necessarily defined in the GML data set. More precisely, the content can be static text specified in the style itself and the text from the GML data set.

Label style has two elements: `gml:style` that specifies the style and `gml:label` that is used to compose the label content. The definitions of the `gml:labelStyle` property and `gml:LabelStyle` descriptor are given in the following listing:

```
<element name="labelStyle" type="gml:LabelStylePropertyType"/>

<complexType name="LabelStylePropertyType">
  <sequence>
    <element ref="gml:LabelStyle" minOccurs="0"/>
  </sequence>
  <attribute name="about" type="anyURI" use="optional"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="LabelStyle" type="gml:LabelStyleType" substitutionGroup="gml:_GML"/>

<complexType name="LabelStyleType">
  <complexContent>
    <extension base="gml:BaseStyleDescriptorType">
      <sequence>
        <element name="style" type="string"/>
        <element name="label" type="gml:LabelTextType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

The `gml:labelStyle` property is defined in the same manner as other GML properties which allow for referencing the value remotely or inline.

The `gml:style` element is used to specify the style of the rendered text. The type of this element is `xsd:string` and the CSS2 styling expressions grammar is used to express graphic properties. The following example shows the use of the `gml:style` element in the geometry style context.

```
<gml:FeatureStyle featureType="exp:City">
  <gml:GeometryStyle>
    <gml:style>fill:blue;stroke:white</gml:style>
  </gml:GeometryStyle>
</gml:FeatureStyle>
```

As noted, the `gml:label` property on the `gml:LabelStyle` descriptor holds the textual content that can be composed of static text and the text extracted from the GML data. The definition of the `gml:label` element that is used to extract the data is given below:

```

<complexType name="LabelType" mixed="true">
  <sequence>
    <element name="LabelExpression" type="string" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute ref="gml:transform" use="optional"/>
</complexType>

```

The content model is mixed to allow both text content and unbounded number of gml:LabelExpression elements. The value of gml:LabelExpression element is an XPath expression that selects the value of some property of the feature.

For example consider this GML data fragment and corresponding gml:label style:

```

<exp:City>
  <gml:name>Belgrade</gml:name>
  <exp:size>1,700,000</exp:size>
  <gml:extentOf>
  ...
  </gml:extentOf>
</exp:City>

<gml:FeatureStyle featureType="exp:City">
  <gml:LabelStyle>
    <gml:style>font-family:Verdana;font-size:18;fill:red</gml:style>
    <gml:label>
      City:
      <gml:LabelExpression>//City/name</gml:LabelExpression>
      , Size:
      <gml:LabelExpression>//City/size</gml:LabelExpression>
    </gml:label>
  </gml:LabelStyle>
</gml:FeatureStyle>

```

This label style will result in the following text being displayed:

City: Belgrade, Size: 1,700,000

## 20.7 Common styling elements

### 20.7.1 Overview

Some common styling elements are used in multiple style descriptors. The gml:symbol element is used by geometry and topology style descriptors. The spatialResolution, styleVariation and animation attributes are declared in gml:BaseStyleDescriptorType, and inherited by geometry, topology, label and graph style descriptors.

### 20.7.2 gml:symbol

The symbol element specifies a graphical symbol used to render a geometry or a topology. A symbol is a description of graphical attributes of a graphical object without a particular, implicit meaning. It can be a description of a line, circle, polygon or more complex drawing. Using the symbol element, we can specify a particular symbol in two ways:

- Remote: Just like any other remote property, the symbol property has the gml:AssociationAttributeGroup attributes that allow for specifying a link pointing to a remote object.
- Inline: The value of the gml:symbol property is the any specifier. This allows for specifying an arbitrary grammar for the symbol.

This element has two additional attributes: symbolType and transform. The symbolType attribute is enumeration and can take one of three values: svg, xpath or other. Applications will rely on the value of this attribute to decide how to interpret the symbol.

The transform attribute allows us to specify a transformation expression that will be applied to the symbol in the rendering phase. Its type is xsd:string and the value is specified in the SVG specification (transform attribute).

### 20.7.3 gml:styleVariation

The function of the styleVariation element is manifold:

- Styling labels: Label style does not have a symbol associated with it since the content is not graphical but is given textually. Using this property we can specify its style attributes.
- Styling symbol variations: One symbol is often used in different cases with slight modifications. It would be cumbersome to create and manage large number of virtually identical symbols; it is easier to create and use only one symbol and express minor differences in its style using this property.
- Parametrized styles: Parametrized styles are styles whose attributes depend on some property of the feature being styled. For example, a city might be styled differently depending on its population. The styleVariation property allows for specifying such dependencies.

The definition of this element's content is:

```
<complexType name="StyleVariationType">
  <simpleContent>
    <extension base="string">
      <attribute name="styleProperty" type="string" use="required"/>
      <attribute name="featurePropertyRange" type="string" use="optional"/>
    </extension>
  </simpleContent>
</complexType>
```

It has two attributes: styleProperty and featurePropertyRange. The value of the styleProperty is an SVG styling attribute name, such as "stroke", "fill", etc. It specifies what attribute of the style the property sets or overrides. The value of the styleVariation element is the value of the styling attribute specified by the styleProperty. The value may be a constant expression or an XPath expression.

The featurePropertyRange attribute defines the subset of features to which the variation applies. Its value is an XPath expression.

The following example shows two variations of the symbol style for a City feature. The feature is styled using a circle symbol. The radius of the circle depends on the population of the city, and is also calculated differently depending whether the population of the city is greater or less than 2 million.

```
<gml:FeatureStyle featureType="exp:City">
  <gml:GeometryStyle>
    <gml:styleVariation
      styleProperty="r"
      featurePropertyRange="population >= 2000000">population div 1000000</gml:styleVariation>
    <gml:styleVariation
      styleProperty="r"
      featurePropertyRange="population < 2000000">population div 1000000</gml:styleVariation>
    <gml:symbol xlink:href="http://www.opengis.org/symbols/City.xml#City"/>
  </gml:GeometryStyle>
</gml:FeatureStyle>
```

#### 20.7.4 **gml:spatialResolution**

The value of the spatialResolution element is defined in the measures.xsd GML schema. The value is derived from the measure type, which is xsd:double type with uom (units of measure) attribute. In GML styling, the meaning of this element is based on the corresponding definition in ISO 19115 (Metadata), where it is defined as a factor that provides a general understanding of the density of spatial data in the data set. Other than this informal definition, GML does not specify the exact use of this attribute. Application developers can use spatialResolution in different ways. For example, it can be used as a map scale denominator (1:50,000, 1:25000, etc.). Applications can also use its value to determine how to draw features in different scales. For example, a city and its features are typically drawn in more details on a large scale map, and perhaps only as a single symbol on a small scale map. Or a coastline can be drawn in detail on a large scale map, while a small scale map application can omit some coordinates for better performance.

#### 20.7.5 **animation**

Animation attributes are used to describe the animation behaviour of the geometry, topology, label or graph. These attributes are defined in the W3C SMIL specification (SMIL 2.0 BasicAnimation Elements):

**Table 6 Attributes used for Animation**

Attribute	Used For
animate	Generic attribute animation
animateMotion	Moving an element along the path
animateColor	Animating colour attributes
set	Setting the value of an attribute for a specified duration

### 20.8 Graph Style

The gml:graphStyle property of the gml:FeatureStyle descriptor has as its value the gml:GraphStyle descriptor which describes style attributes of a graph formed by a set of features. The definitions of the graph style property and descriptor are shown in the following listing:

```
<element name="graphStyle" type="gml:GraphStylePropertyType"/>

<complexType name="GraphStylePropertyType">
  <sequence>
    <element ref="gml:GraphStyle" minOccurs="0"/>
  </sequence>
  <attribute name="about" type="anyURI" use="optional"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<element name="GraphStyle" type="gml:GraphStyleType" substitutionGroup="gml:_GML"/>

<complexType name="GraphStyleType">
  <complexContent>
    <extension base="gml:BaseStyleDescriptorType">
      <sequence>
        <element name="planar" type="boolean" minOccurs="0"/>
        <element name="directed" type="boolean" minOccurs="0"/>
        <element name="grid" type="boolean" minOccurs="0"/>
        <element name="minDistance" type="double" minOccurs="0"/>
        <element name="minAngle" type="double" minOccurs="0"/>
        <element name="graphType" type="gml:GraphTypeType" minOccurs="0"/>
        <element name="drawingType" type="gml:DrawingTypeType" minOccurs="0"/>
        <element name="lineType" type="gml:LineTypeType" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

<element name="aestheticCriteria" type="gml:AestheticCriteriaType" minOccurs="0"
maxOccurs="unbounded"/>
  </sequence>
</extension>
</complexContent>
</complexType>

```

The gml :graphStyle property is defined in the same manner as other GML properties which allow for referencing the value remotely or inline.

Graph style descriptor describes the style for a graph as a whole, not for individual graph elements. It inherits from the base content model common styling properties described in the section 20.2.

This descriptor adds to the base content model a group of properties specific to graph styling – they describe the graph in terms of its specific characteristics. The properties are described in the following table.

**Table 7 Elements used in graph styling**

Element	Type	Use
Planar	xsd:boolean	If true, the graph edges do not cross (planar graph); if false they may cross
directed	xsd:boolean	If true the graph is directed; if false it is not directed
Grid	xsd:boolean	If true, the coordinates of vertices, crossings and bends have integer values, otherwise they may have decimal values
minDistance	xsd:double	A recommendation for the minimum distance between vertices and non-incident edges
minAngle	xsd:double	A recommendation for the minimum angle between consecutive incident edges (angular resolution)
graphType	xsd:enumeration	The type of the graph. The value may be TREE or BICONNECTED
drawingType	xsd:enumeration	the type of the drawing with respect to the orthogonality of edges. The value may be POLYLINE or ORTHOGONAL
lineType	xsd:enumeration	Determines whether there will be any bent edges. The value may be STRAIGHT or BENT
aestheticCriteria	xsd:enumeration	A recommendation for the general outline of the graph according to a particular aesthetic criteria. The value may be one of the following: MIN_CROSSINGS, MIN_AREA, MIN_BENDS, MAX_BENDS, UNIFORM_BENDS, MIN_SLOPES, MIN_EDGE_LENGTH, MAX_EDGE_LENGTH, UNIFORM_EDGE_LENGTH, MAX_ANGULAR_RESOLUTION, MIN_ASPECT_RATIO or MAX_SYMMETRIES

## 21 Modularisation and Dependencies

The base schemas for GML described above have been modularised so that application schemas that do not need the complete set of GML definitions may import only the topical subsets of GML that are required. For example, a GML version 2 application schema migrating to GML version 3 without adding any new definitions could continue to import feature.xsd. It would not import or require parsing of the new GML version 3 definitions for coordinate reference systems, topology, coverages, dynamic features, default styles, and observations. However, it would import and require parsing of all of the basic types that have been added in GML version 3.

The modularisation of GML creates the dependencies among the GML base schemas shown in Figure 11 below. A dashed arrow in the figure indicates that the schema at the tail of the arrow depends upon the schema at the head of the arrow. A dependency may occur where one schema <include>s another schema in the “gml” namespace. For example, feature.xsd <include>s geometryBasic2d.xsd. A dependency may also occur where one schema <import>s another schema for a namespace other than “gml”. For example, gmlBase.xsd <import>s xlink.xsd from the “xlink” namespace.

There are now seven schemas in GML upon which no other GML schemas depend. These top level schemas are the roots of partially overlapping hierarchies of GML schemas:

- observation.xsd
- dynamicFeature.xsd
- coverage.xsd
- topology.xsd
- defaultStyle.xsd
- coordinateReferenceSystems.xsd
- temporalReferenceSystems.xsd

An application schema that needs definitions from more than one of these GML topical subset schema hierarchies can <import> gml.xsd and get all of the GML definitions. Or it can contain multiple <import>s for just the appropriate gml schema documents, thereby excluding unwanted GML type definitions. However, as specified in [XMLSchema-2] clause 4.2.3, “it is open to applications to ignore all but the first <import> for a given namespace”.

To work around this problem, an application schema writer may create a custom top-level GML schema by copying the <schema> element from gml.xsd, and <including> just the appropriate gml schema documents. This custom top-level GML schema in the “<http://www.opengis.net/gml>” namespace named “gml” is then <import>ed into the application schema, which uses its own application-specific namespace. For example, an application schema for features with topology could <import> a custom top-level GML schema that <include>s just feature.xsd and topology.xsd, thereby importing 17 fewer schemas than would have been imported using gml.xsd.

However, when an application schema will be used in a processing environment that lacks CPU, memory and/or I/O bandwidth, for example, in a mobile hand-held device, an absolutely minimal <import> of GML definitions is often desired. The custom top-level GML schema approach described above might bring in a unacceptably large number of unwanted definitions from each GML schema <include>d in the custom top-level GML schema. The solution is to create a single GML subset schema that contains exactly the required GML type and element definitions. However, creating such a GML subset schema by hand using a text or XML editor to cut and paste definitions is a tedious and error-prone process because it involves analyzing type definition dependencies across the many GML schema documents. An automated approach is recommended instead. An informative sample implementation of a GML schema subset tool is included in Annex F. Subset schemas, however they are produced, are Profiles of GML as described in Clause 22.

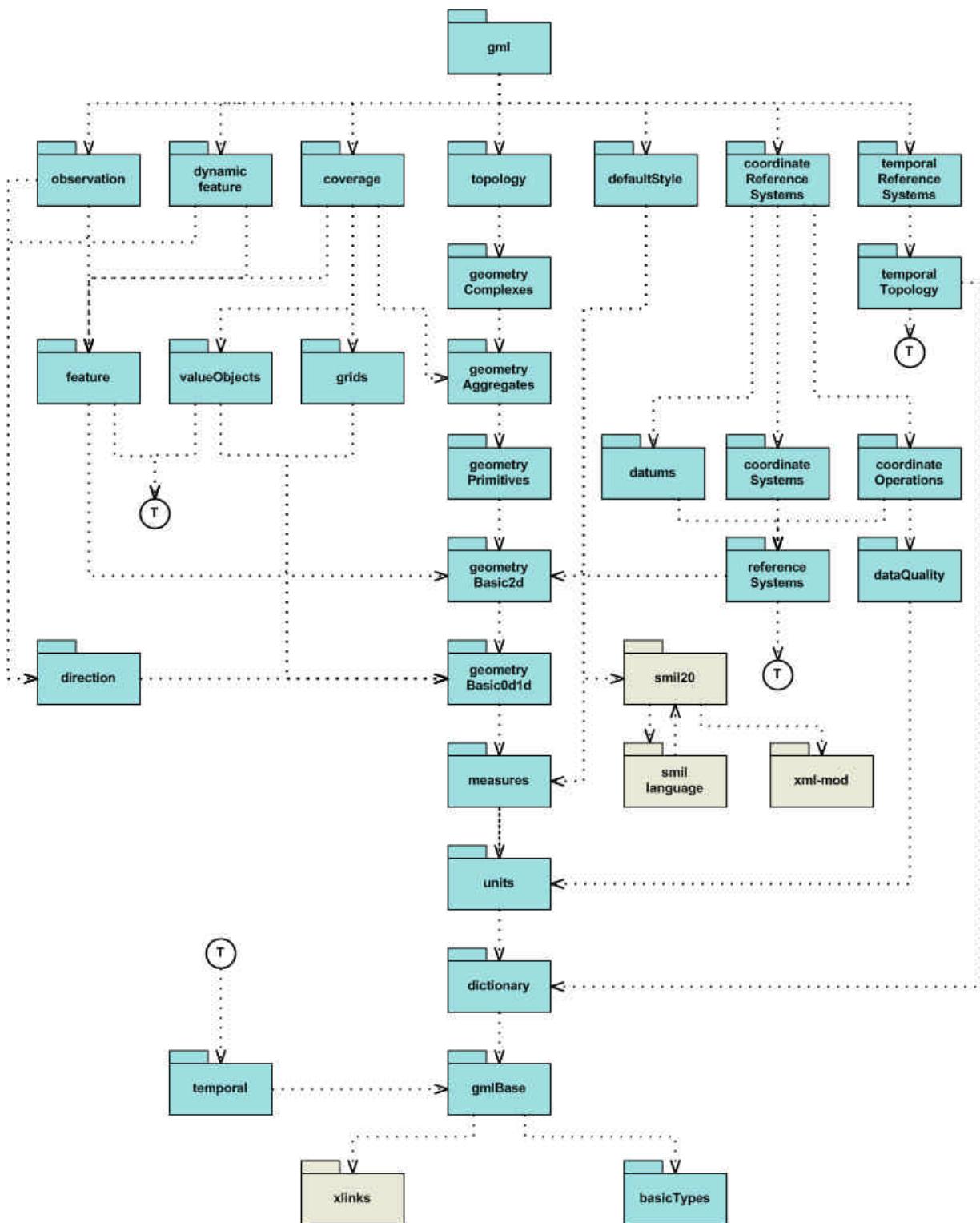


Figure 11 — Schema Dependencies

## 22 Profiles

### 22.1 Profiles of GML and application schemas

GML is a complex specification that is richly expressive. In general, an application need not exploit the entire specification, but may employ a subset of constructs corresponding to specific relevant requirements.

Quoting from technical report “ISO/IEC TR 10000-1:1998 Information technology - Framework and taxonomy of International Standardized Profiles - Part 1: General principles and documentation framework”, we use this definition of a profile:

**Profile:** *A set of one or more base standards and/or [profiles], and, where applicable, the identification of chosen classes [(types, attributes and elements)], conforming subsets, options and parameters of those base standards, or [profiles] necessary to accomplish a particular function.*

Of course, this was defined for an OSI architecture model, so we must translate ‘class’ to ‘types, attributes and elements’ to apply this definition to XML Schema. There are several ways to implement this, and GML profiles uses a “copy and delete” approach. To create a profile, a developer might copy the applicable schema files from GML and simply delete any global types, elements and local optional particles that she does not need for her application schema.

### 22.2 Definition of Profile

A profile of GML can be defined to enhance interoperability and to curtail ambiguity by allowing only a specific subset of GML. Application schemas can then conform to such a profile in order to take advantage of any interoperability or performance advantages that it offers in comparison with a complete GML. Such profiles can be defined for application schemas that are included in other OGC specifications.

There are cases where reduced functionality is acceptable, or where processing requirements compel use of a logical subset of GML. For example, applications that do not need to handle XLink attributes in any form can adhere to a specific profile that excludes them; the constraint in this case would be to not use links. Other cases might include defining constraints on the level of nesting allowed inside tags (i.e. tree depth), or only allowing features with homogeneous properties as members of a feature collection. In many cases, such constraints can be enforced via new schemas; others may be enforced through procedural agreements within an information community.

### 22.3 Relation to application schema

A profile can be the beginning of an application schema. For example, a location based service profile can limit the types of geometry to that used in LBS applications, and the LBS application schema can then add a “PointCircle,” “PointEllipse” and “PointArc” elements to accommodate the LIF “CIRCLE,” “ELLIPSE” and “ARC” elements, which are used to describe error estimates of mobile device location.

The building of such application schemas is thus a two-part process. The profile acts as a restriction of GML to produce types and elements consistent with the complete GML 3 but potentially lacking in some optional particles. The application schema then uses these types as a common base, and uses them in new types and elements by extensions or inclusion.



### 22.4 Rules for elements and types in a profile

Global profiled elements in a GML profile shall:

- Share the same name (and namespace) of a parent element in GML.
- Include all mandatory particles (subelements and attributes) of the parent element in GML.

- c) Include no particle that is not in the parent element in GML.
- d) Have the same default values for attributes as the parent element in GML.
- e) Have a parallel substitution group hierarchy for named elements in both schemas.

Global types in a GML profile shall:

- a) Share the same name (and namespace) of a parent type in GML.
- b) Include all mandatory particles (subelements and attributes) of the parent type in GML.
- c) Include no particle that is not in the parent type in GML.
- d) Have the same default values for attributes as the parent type in GML.
- e) Have a parallel derivation tree for named types in both schemas.

Instance documents of a profile shall be valid against the full GML schema.

Using the “copy and delete” metaphor described above, our mythic developer can:

- a) Delete global element and global types.
- b) Delete optional subelements from any types or elements
- c) Make optional subelements or attributes mandatory in any type or element (if a default value exist, it shall be eliminated or the schema validation will report an error — default values are only valid for optional particles)
- d) Restrict cardinality of any particle.

None of the above will affect the validity of a document that is designed against the profile, but tested against the full GML schema. Our mythic developer **cannot**:

- a) Delete mandatory subelements from any types or elements.
- b) Make mandatory particles optional.
- c) Relax cardinality restrictions of any particle.
- d) Add or change a default or fixed value.

Item d) is a bit subtler than the others are. Documents valid under the profile would still be valid under the full GML schema, but the interpretation of those documents would change. For example, if a profile specified a default coordinate reference system to be UTM, and the full schema specified a WGS84 geodesic (latitude, longitude) as the default CRS, then the interpretation of the file would change when moving from the profile to the full schema.

## 22.5 Recommendations for application schemas using GML profiles

In order that the profile within an application schema can be later extended to include other profiled GML elements, the following recommendations are made:

- a) Global elements that are not in a GML profile but are in an application schema using a GML profile should not have the same name as any element in the GML schemas.

- b) Global types that are not in a GML profile but are in an application schema using a GML profile should not have the same name as any type in the GML schemas.

If a type or element in an application schema is found to be of universal use, then the above conventions will aid the application schema from migrating that type or element from its own namespace to that of GML.

The following recommendations are made simply as a bookkeeping convenience to those trying to understand the role of the profile in the application schema:

- a) Profiled elements and types should be included in a file structure that parallels that of GML. The exact naming convention of the parallelism is left to the application schema author.
- b) A reference to the appropriate GML schema file should be made in a comment near the beginning of the file.
- c) The profile should use the GML namespace (<http://www.opengis.net/gml>)

## 22.6 Summary of Rules for GML profiles

In summary, the rules for a profile:

- a) A profile of GML is a logical restriction of a subset of GML.
- b) A profile shall not change the name, definition, or data type of mandatory GML elements or attributes.
- c) The relevant schema or schemas that define a profile shall use in the core 'gml' namespace <http://www.opengis.net/gml>.
- d) An application schema may extend and use types from the profile, but shall do so in its own namespace, and not use <http://www.opengis.net/gml>.

The functional test of these rules is:

In any instance document for an application schema using a GML profile will be valid against the same application schema if the GML profile is replaced by the complete GML schema. Further, the interpretation of that document would be the same regardless of which of the two schemas were used.

# 23 Rules for Application Schemes

## 23.1 GML Documents

An XML document contains a single XML element as its root. Several document types are permitted using GML, including:

- A Feature
- A Feature Collection
- A Coverage (Coverages are Features so is a Feature Collection)
- A Dictionary, including a Coordinate Reference System Dictionary or Units of Measure Dictionary
- A Topological Complex

The standard methods for XML documents based on W3C XML Schema, provide that the XML namespaces used in a document are declared as attributes within the document, and the location of schema documents that provide the source components for each namespace may be indicated.

For a GML document, the source of the components describing the primary components within the document is a GML Application Schema. Both the document type and the associated application schema are described in this clause.

## 23.2 GML Application Schemas

A GML Application Schema is an XML Schema, conforming to the rules outlined in this clause, that describes one or more types of geographic objects, components of geographic objects, or meta data, including dictionaries and definitions, used in the definition of geographic objects.. A GML Application Schema defines a vocabulary for a particular domain of discourse by defining and describing the terms of that vocabulary (see ISO 19109) as follows.

An application schema directly uses concrete GML elements and attributes whose names and content models accurately represent components of the vocabulary it defines, e.g. `gml:location`, `gml:name`, `gml:description`, `gml:Observation`, `gml:id`, `gml:Dictionary`, `gml:FeatureCollection`.

An application schema declares new elements and attributes in its own namespace using GML types when the vocabulary it defines needs to include different names for the same content models to distinguish their semantic roles. The element declared in the application schema will be in a different namespace, and may be used in an instance document. E.g. `gml:EnvelopeType` may be used unmodified as the content model for an element `xmml:Interval`.

An application schema derives new types in its own namespace by extension of GML types when the vocabulary it defines needs to include components with additional, domain-specific properties.

An application schema derives new types in its own namespace by restriction of GML types when the vocabulary it defines needs to include more specialized versions of GML types that restrict the cardinality or type of their properties.

An application schema declares new elements and attributes in its own namespace using types it has defined to give vocabulary-specific names to their content models.

An application schema declares new elements that are assigned to a substitution group whose head is an abstract or concrete GML element. The element declared in the application schema may then appear in instance documents in place of the substitution group head and be conformant to the content model that refers to the substitution group head. Note that in order to be a valid member of a substitution group, the type of the element shall be validly derived from the type of the element which is the head of the substitution group. All abstract elements in the GML schemas are only useful acting as the heads of substitution groups. E.g. `gml:_GML`, `gml:_Feature`, `_gml:_Geometry`, `gml:_TimeObject`

All GML Application Schemas are constructed, using the general rules of this clause and its sub clauses, from one or more of the core GML schemas defined in Clauses 7 to 20.

In GML 2, GML Application Schemas were restricted to the development of vocabularies of geographic features and all application schemas were based on the `feature.xsd`.

In GML 3, this concept is extended and several different types of application schemas can be constructed including Feature Schemas, Geometry Schemas, Topology Schemas, Temporal Schemas, Coordinate System Schemas, Coverage Schemas, Observation Schemas, and Value Schemas. Additional rules for the construction of schemas are discussed in the following clauses.

GML also allows the derivation of many other kinds of elements such as new units of measure, new geometry properties and new geometries. While these elements can be packaged into separate schemas they are viewed as subordinate to the schema categories of this clause. Any conformant GML application schema shall be one of the schema types listed below, or be a schema document that complies to the rules from the

respective schema clause. It is thus permissible to create a GML application schema that defines Features, Coverages and Values, so long as this schema satisfies the rules of Clauses 23.3, 23.8 and 23.9.

### **23.2.1 Target Namespace**

An application schema shall declare a target namespace. This is the namespace in which the terms for objects and properties of the vocabulary defined by the application schema “live”. This shall not be the GML namespace (<http://www.opengis.net/gml>). It is conventional for the namespace identifier to be a URL controlled by the application schema author’s organization. A target namespace is declared in an application schema using the targetNamespace attribute of the schema element from XML Schema.

An application schema may be comprised of multiple schema documents that all declare the same target namespace. Note that a top level schema document in such a modularised application schema should directly or indirectly include the others to avoid the XML application processing limitatations discussed in section 21 Modularisation and Dependencies above.

### **23.2.2 Import GML Schema**

An application schema shall import all of the components from GML 3 that it directly or indirectly uses to define its vocabulary. The required import of a specific gml schema in clauses 23.\*.2 below may be provided indirectly via the import of another schema in the namespace for gml that includes the required gml schema. For example, the import of gml.xsd would satisfy any of these schema import requirements. In addition, the required import of a gml schema may be provided by the import of an equivalent subset schema as described in clause 22, or by the import of an equivalent schema from a GML profile that defines the complex type of the document root element. These are all equivalent schemas with respect to satisfying the schema import requirements. For example:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="../gml.xsd"/>
```

Note that the <import> element specifies that the components described in gml.xsd are associated with the GML namespace <http://www.opengis.net/gml>. This namespace identifier shall match the target namespace specified in the schema being imported in order to ensure XML Schema validity.

The path (schemaLocation) to the imported GML schema can be to a local copy of the document, or may be a URI reference to a copy of the schema document in some remote repository, such as the repository <http://schemas.opengis.net/> on the OGC web site.

### **23.2.3 Object Type Derivation**

An object type declared by an application schema shall not violate any XLM Schema derivation restrictions imposed by a “final” attribute on its base GML type.

An object type defined by an application schema shall derive directly from the most specialized GML object type that can serve as the base for its content model while preserving semantic consistency and increasing type specialization.

### **23.2.4 Elements Representing Objects**

An application schema shall declare a global element for any object type that is to serve as the root element in a GML instance document.

### **23.2.5 Property Type Derivation**

A property type defined by an application schema to contain or reference a single GML object may be derived from gml:AssociationType, or may follow the pattern of gml:AssociationType.

A property type defined by an application schema to contain or reference a homogeneous collection of GML objects shall be derived from `gml:ArrayAssociationType`.

### 23.2.6 Elements Representing Properties

Elements representing properties of GML objects may be declared as global elements in an application schema, or they may be declared locally within object content models (type definitions). Note that elements in the content of complex types that are defined with local names in an application schema will prevent derivation by restriction in another namespace. Such complex types are appropriate for elements intended for use "as is" in their own namespace, and should be declared to be `final="restriction"`. Elements in the content of complex types defined by reference to global elements support derivation by restriction in another namespace, allowing restriction of cardinality, and/or replacement by a member of a substitution group. Such complex types designed for derivation by restriction are appropriate "library types" for elements in substitution groups that cross namespaces.

If the value of the property is expected to be available elsewhere, it is necessary for the property element in an instance to provide a pointer to the value, so the property shall support association by-reference. This is accomplished using `xlink` attributes, and in the GML application schema the XML type for the property shall include `gml:AssociationAttributeGroup`.

If the value of the property is expected to be represented in place, the content model (XML schema type) for the property shall support this. The property element shall either have XML Schema `simpleContent` of the appropriate `simpleType`, or it shall have `complexContent` comprising a **single** child element of the appropriate type.

If the value of the property is expected to be available either elsewhere, or represented in place, then the type for the property element shall support both methods. The type for the property element shall have the `gml:AssociationAttributeGroup`, in which all members are optional, and the child element shall have `minOccurs="0"` so that in an instance document the property element may be empty if it carries an `xlink`. Note that the value of a property of an `ArrayAssociationType` shall be represented in place.

If the last pattern is used, and it is desired to prohibit the possibility of both `xlink` attributes and content, or neither, then this constraint should be recorded as a normative directive in an `<annotation>` element on the element declaration in the application schema. The directive may be expressed as prose, or it may be expressed using a formal notation such as Schematron.

## 23.3 Schemas defining Features and Feature Collections

Features and Feature collections are the primary view of geospatial information supported by GML, and are particularly useful in modeling real world geography or in defining message types for geographic web services. A Feature in GML models a real world object or concept and provides an element (feature type) and an associated set of properties describing that type also encoded as elements.

Feature application schemas define geographic features and feature collections for a specific application domain or community . These application schemas shall obey the additional rules described in the following clauses.

### 23.3.1 Target Namespace

The application schema shall declare a target namespace as described in section 22.3.1 Target Namespace, in which the terms (features, feature collections) of the vocabulary "live".

### 23.3.2 Import feature.xsd

The application schema shall import the necessary components from GML 3 as described in section 22.3.2 Import GML Schemafor example as follows:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="../feature.xsd"/>
```

### **23.3.3 Feature Type Derivation**

All feature types declared in an application schema shall derive either directly or indirectly from `gml:AbstractFeatureType`.

The abstract type `gml:AbstractFeatureCollectionType` may be used as the base type for feature collection types. Alternatively, feature collection types may be specified following the pattern used in this abstract type (directly or indirectly derive from `gml:AbstractFeatureType` and contain a feature-valued property).

### **23.3.4 Elements Representing Features**

All geographic features and feature collections in the application schema shall be declared as global elements in the schema, i.e. they shall be immediate child elements of the XML Schema `<schema>` element.

The name of an element that instantiates a GML feature shall be its Feature Type, in the sense described in ISO 19109 and 19110.

### **23.3.5 Application Features are Features**

A feature defined in an application schema shall conform to the rules respecting GML features as described in Clause 8. These include in particular:

- The name of feature element is the semantic type of the feature.
- The children of a feature are always properties that describe the feature, and such properties can only be encoded as child elements. Properties cannot be encoded as XML attributes.

### **23.3.6 GML Feature Collection Document**

A GML application may use a `gml:FeatureCollection` to contain features defined in an application schema when there is no requirement to restrict the inclusion of other types of features. A GML Application Schema may define specialised feature collection types and global elements to serve as the root elements for documents describing Feature Collections specific to the application domain where there is a requirement to exclude features not defined in that domain.

## **23.4 Schemas defining Geometries**

### **23.4.1 Target Namespace**

The application schema shall declare a target namespace as described in section 22.3.1 Target Namespace, in which the terms (geometries) of the vocabulary “live”.

### **23.4.2 Import a GML geometry schema**

The application schema shall import the necessary components from GML 3 as described in section 22.3.2 Import GML Schema, for example as follows:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="..../geometryComplexes.xsd"/>
```

### 23.4.3 User-defined Geometry Types and Geometry Property Types

#### 23.4.3.1 User-defined Geometry Types

Authors of application schemas may create their own geometry types if GML lacks the desired construct. To do this, authors shall ensure that these concrete geometry and geometry collection types are subtyped (either directly or indirectly) from the corresponding GML type: AbstractGeometryType.

**EXAMPLE** The following complex type definition in an application schema extends the PointType of GML and adds a bearing (e.g. for the orientation of a symbol in portrayal).

```
<complexType name="PointWithBearingType">
  <complexContent>
    <extension base="gml:PointType">
      <sequence>
        <element name="bearing" type="gml:AngleType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Any user-defined geometry subtypes shall inherit the elements and attributes of the base GML geometry types without restriction, but may extend these base types to meet application requirements, such as providing a finer degree of interoperability with legacy systems and data sets.

#### 23.4.3.2 User-defined Geometry Property Types

Furthermore, authors of application schemas may create their own geometry property types that encapsulate geometry types they have defined according to clause 9.1. They shall ensure that these properties follow the pattern used by gml:Geometry.PropertyType. For standard properties and gml:GeometryArray.PropertyType for array properties. The target type shall be a bonafide geometry construct.

A geometry property type may be a restriction of Geometry.PropertyType, but this is not a requirement. Nevertheless, every geometry property shall follow the pattern of this type. It is allowed to support the choice between a by-value or a by-reference semantic or to restrict the use to either by-value (prohibit the use of the Xlink attributes) or by-reference (prohibit the containment of the geometry in the feature).

A geometry array property type may be a restriction of GeometryArray.PropertyType, but this is not a requirement. Nevertheless, every geometry property shall follow the pattern of this type. All geometry elements in the array are contained in the feature, only by-value semantics is supported by array properties.

**EXAMPLE** The following complex type definitions in an application schema define a “standard” property type for an user-defined geometry type and an array property type for the same geometry type.

```
<complexType name="MyGeometry.PropertyType">
  <sequence>
    <element ref="foo:PointWithBearingType" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup" />
</complexType>

<complexType name="MyGeometryArray.PropertyType">
  <sequence>
    <element ref="foo:PointWithBearingType" minOccurs="0" maxOccurs="unbounded" />
  </sequence>
</complexType>
```

## **23.5 Schemas defining Topologies**

### **23.5.1 Target Namespace**

The application schema shall declare a target namespace as described in section 22.3.1 Target Namespace, in which the terms (topologies) of the vocabulary “live”.

### **23.5.2 Import topology.xsd**

The application schema shall import the necessary components from GML 3 as described in section 22.3.2 Import GML Schema, for example as follows:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="../topology.xsd"/>
```

### **23.5.3 User-defined Topology Types and Topology Property Types**

#### **23.5.3.1 User-defined Topology Types**

Authors of application schemas may create their own topology types if GML lacks the desired construct. To do this, authors shall ensure that these concrete topology types are subtyped (either directly or indirectly) from the corresponding GML type: AbstractTopologyType.

Any user-defined topology subtypes shall inherit the elements and attributes of the base GML topology types without restriction, but may extend these base types to meet application requirements, such as providing a finer degree of interoperability with legacy systems and data sets.

#### **23.5.3.2 User-defined Topology Property Types**

Furthermore, authors of application schemas may create their own (directed) topology property types that encapsulate topology types they have defined according to section 13. They shall ensure that these properties follow the pattern used by the gml property type for the base gml Topology type, e.g. gml:DirectedNodePropertyType, or gml:TopoVolumePropertyType, or gml:TopoComplexMemberType. The target type shall be a bonafide topology construct.

A topology property type may be a restriction of a Topology PropertyType, but this is not a requirement. Nevertheless, every topology property shall follow the pattern of this type.

## **23.6 Schemas defining Time**

### **23.6.1 Target Namespace**

The application schema shall declare a target namespace as described in section 22.3.1 Target Namespace, in which the terms of the vocabulary “live”.

### **23.6.2 Import temporal.xsd**

The application schema shall import the necessary components from GML 3 as described in section 22.3.2 Import GML Schema, for example as follows:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="../temporal.xsd"/>
```

### 23.6.3 User-defined Temporal Types and Temporal Property Types

#### 23.6.3.1 User-defined Temporal Types

Authors of application schemas may create their own temporal types if GML lacks the desired construct. To do this, authors shall ensure that these concrete temporal types are subtyped (either directly or indirectly) from the corresponding GML type: AbstractTimeObjectType.

Any user-defined temporal subtypes shall inherit the elements and attributes of the base GML temporal types without restriction, but may extend these base types to meet application requirements, such as providing a finer degree of interoperability with legacy systems and data sets.

#### 23.6.3.2 User-defined Temporal Property Types

Furthermore, authors of application schemas may create their own temporal property types that encapsulate temporal types they have defined according to section 13. They shall ensure that these properties follow the pattern used by the `gml:propertyType` for the base GML temporal type, e.g. `gml:TimeInstantPropertyType`. The target type shall be a bonafide temporal construct.

A temporal property type may be a restriction of a `TemporalPropertyType`, but this is not a requirement. Nevertheless, every temporal property shall follow the pattern of this type.

## 23.7 Schemas defining Coordinate Reference Systems

Many of the concrete XML elements defined in the CRS Schemas can be used without Application Schemas, whenever no content extensions or restrictions are needed. An Application Schema shall be used whenever element contents extension is required, and should be used in most other cases to specify needed restrictions. That is, an Application Schema should be defined to extend and/or restrict elements as needed for a specific application, or a set of applications, to:

- a) Add elements to contents of existing elements, for recording additional data about that item needed for that application.
- b) Restrict the multiplicity of current contents elements, to eliminate flexibility not needed and perhaps confusing for that application.
- c) Use a different element name, to be more easily understood in that specific application, primarily for elements that will be instantiated many times.
- d) Specify standard contents and contents patterns for selected elements and attributes, as needed to improve interoperability.
- e) Specify standard XML and other documents to be referenced or otherwise used, as needed to improve interoperability.

Application Schemas can thus be used for XML document contents extensions, restrictions, or both. Contents extension is expected to be often used to record additional data needed for applications. Contents restriction is expected to be frequently used to restrict contents, in order to increase interoperability and reduce ambiguity when greater flexibility is not needed for applications. Extensions of existing concrete elements can be defined by extending that concrete element. In many cases, restrictions of existing concrete elements can be done by extending the abstract element from which that concrete element is derived, by adding somewhat different but corresponding extensions.

An Application Schema can specify a single top level element for use in an XML document, with the XML elements and types that it uses. That single top level XML element can be an object with identity, but this is not required. Such an Application Schema will import and build upon one or more of the six XML Schemas specified in this document.

Application Schemas could define additional concrete elements using or extending other abstract elements, if needed. However, an additional concrete element using or extending an abstract element should not be defined if that concrete element is largely similar to an existing element, and thus probably should extend or use an existing concrete element. In many cases, the existing concrete elements that use an abstract element are believed to be largely exhaustive. This is particularly true when the existing concrete elements include one element that is quite general, such as the elements EngineeringCRS, DerivedCRS, EngineeringDatum, UserDefinedCS, OperationParameter, and OperationParameterGroup.

The Conversion, Transformation, ParameterValue, and ParameterValueGroup elements can be used for well-known coordinate operation methods, especially when only one instance of that element is needed for that operation method. However, these elements should not be used for well-known coordinate operation methods when many instances of that element are needed for one operation method. Instead, an Application Schema that defines operation-method-specialized element names and contents should be prepared for each such operation method. Subclauses F.3 and F.4 provide examples of such Application Schemas. For interoperability, a suitable geospatial information community should standardize each such Application Schema.

NOTE This use of Application Schemas follows the patterns used in Feature Application Schemas.

### 23.7.1 Target Namespace

The application schema shall declare a target namespace as described in section 22.3.1 Target Namespace, in which the terms (coordinate reference systems) of the vocabulary “live”.

### 23.7.2 Import coordinateReferenceSystem.xsd

The application schema shall import the necessary components from GML 3 as described in section 22.3.2 Import GML Schema, for example as follows:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="../coordinateReferenceSystem.xsd"/>
```

## 23.8 Schemas defining Coverages

Coverages are an alternative view of geospatial information. This view focuses on the variation of a property or properties across a domain, so is particularly useful in analysis.

This clause defines the rules for the construction of application schemas for coverages. Note that coverages are features and hence the rules of section 23.3 above apply also to coverages.

### 23.8.1 Target Namespace

The application schema shall declare a target namespace as described in section 22.3.1 Target Namespace, in which the terms (coordinate reference systems) of the vocabulary “live”.

### 23.8.2 Import coverage.xsd

The application schema shall import the necessary components from GML 3 as described in section 22.3.2 Import GML Schema, for example as follows:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="../coverage.xsd"/>
```

### 23.8.3 Coverages shall derive from gml:AbstractDiscreteCoverageType or gml:AbstractContinuousCoverageType

All geographic coverages in the application schema shall be declared as global elements in the schema, that is they shall be child elements of the XML Schema `<schema>` element. The content model for such global elements shall derive by extension either directly or indirectly from `gml:AbstractDiscreteCoverageType` or

gml:AbstractContinuousCoverageType. Note that these types derive indirectly from gml:AbstractFeatureType and hence the condition of the feature model is satisfied.

The coverage.xsd provides the specific coverage types (MultiPointCoverage, MultiCurveCoverage, MultiSurfaceCoverage, MultiSurfaceCoverage, GriddedCoverage, RectifiedGridCoverage) and application coverages can derive from any of these as well.

#### 23.8.4 Range Parameters shall be Derived from gml:ValueType

The coverage application schema shall define or import the definitions for all Range Parameters. Each such Range Parameter shall be substitutable for gml:\_Value as defined in the valueObjects.xsd schema. Note that this allows the Range Parameter to take on a wide range of types. Note further that the value.xsd provides several abstractsub- sub-types that are substitutable for gml:\_Value, including gml:\_ScalarValue and gml:\_ValueList. Concrete scalar and value list types, and substitution group head elements, are also provided (substitutable for gml:\_ScalarValue or gml:\_ValueList) and include:

- gml:Category (gml:CodeType)
- gml:CategoryList (gml:CodeOrNullListType)
- gml:Quantity (gml:MeasureType)
- gml:QuantityList (gml:MeasureOrNullListType)
- gml:Count (gml:CountType)
- gml:Boolean (gml:BooleanType)

To define the Range Parameters in a Coverage Application schema, refer to the valueObjects.xsd schema described in Annex C.

Typical examples of the use of the value types in the development of a GML coverage can be found in the examples clause for coverages Clause 19.3, and are summarized in Table 8.

**Table 8 – Range Parameters for Coverage Schemas**

Coverage	Range Parameter	Definition in GML
Temperature Distribution (weather)	Temperature	Would be derived from gml:MeasureOrNullListType and made substitutable for gml:measure defined in the measures.xsd.
Soil type distribution (agronomy)	Soil type	Would be derived from gml:Category and made substitutable for gml_Category. Weak reference to an enumeration of soil types.
Multi-spectral optical image (remote sensing)	Reflectance in each spectral band.	Would be derived from gml:QuantityListType and made substitutable for gml:QuantityList.
Distribution of West Nile Virus cases. (epidemiology)	CaseCount	Would be derived from integerOrNullList, and made substitutable for gml:CountList..
Distribution of West Nile Virus cases. (epidemiology)	CaseCount	Would be derived from integerOrNullList, and made substitutable for gml:CountList.

### **23.8.5 Coverage Document**

A coverage document is defined by a corresponding coverage schema. The root element of this document shall be a coverage defined in this schema or may be a feature collection whose members are coverages as described in Clause 8.2.7.

## **23.9 Schemas defining Observations**

An observation in GML is a kind of feature that represents an act of observing or observation event. This may be associated with a measurement obtained from some type of instrument, or may just be a photograph acquired by a travelling tourist.

This clause describes how to create an observation application schema.

An observation application schema defines one or more types of observations according to the following rules.

### **23.9.1 Target Namespace**

The application schema shall declare a target namespace as described in section 22.3.1 Target Namespace, in which the terms (observation) of the vocabulary “live”.

### **23.9.2 Import observation.xsd**

The application schema shall import the necessary components from GML 3 as described in section 22.3.2 Import GML Schema, for example as follows:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="../observation.xsd"/>
```

### **23.9.3 Observations shall derive from gml:ObservationType**

All observation types (kinds of observations) defined in the application schema shall be declared as global elements in the schema, that is they shall be child elements of the XML Schema `<schema>` element. The content model for such global elements shall derive by extension either directly or indirectly from `gml:ObservationType`.

### **23.9.4 Observation Collections shall derive from gml:CollectionType**

All observation collections in the application schema shall be declared as global elements in the schema, that is they shall be child elements of the XML Schema `<schema>` element. The content model for such global elements shall derive by extension either directly or indirectly from `gml:CollectionType`.

### **23.9.5 Observations are Features**

An observation defined in an application schema shall conform to the rules respecting GML features as described in Clause 8. These include in particular:

- a) Observation semantic type information can ONLY be carried by elements. The following is not valid GML.

```
<abc:Measurement type ="WaterQuality"> ... </abc:Measurement>
```

- b) The children of a observation are always properties that describe the observation, and such properties can only be encoded as child elements. Properties cannot be encoded as XML attributes.

- c) No child of an observation can itself be a feature or other GML object (i.e. one derived from `gml:AbstractGMLType`). Thus no child of an observation can be feature, geometry, topology, coordinate reference system etc.

### 23.9.6 Observation Collection Document

Corresponding to an Observation Collection document there shall be a GML Application Schema that defines the single root element of that Observation Collection document. Note that this does not imply that this Observation Collection is defined by a single application schema. The features referenced from the Observation Collection element may be contained in any number of other schemas and these may define observations only, observation collections or any combination of the same.

An observation collection could be used, for example, to encode a set of measurements from one or more sensor devices.

## 23.10 Schemas defining Dictionaries and Definitions

A common requirement is to collect a set of definitions together into a dictionary, in order that a term may be referred to many times, while its (potentially lengthy) definition is only recorded once. The kind of information that might form the content of a dictionary are units of measure, coordinate reference systems, observable-types or measurands, parties including individuals or organizations.

In order to support this, some generic components for dictionaries and definitions are provided in GML 3. The basic Definition is a simple element which supports a prose description. This may be used directly for simple non-parameterised definitions, or may serve as the basis for specialized definition elements.

The basic Dictionary is a bag of Definitions. It may be used as a container for an arbitrary set of Definitions or elements that are in the Definition substitution group. It may also serve as the basis for a specialized Dictionary restricted to contain only certain types of definition.

One set of specialized definitions is built in to GML 3, for units of measure, and serves as an example of how to derive specialized definition components.

This clause describes how to create an application schema for definitions.

An application schema for definitions defines one or more types of definitions according to the following rules.

### 23.10.1 Target Namespace

The application schema shall declare a target namespace as described in section 22.3.1 Target Namespace, in which the terms (dictionaries, definitions) of the vocabulary “live”.

### 23.10.2 Import dictionary.xsd

The application schema shall import the necessary components from GML 3 as described in section 22.3.2 Import GML Schema, for example as follows:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="../dictionary.xsd"/>
```

### 23.10.3 Definitions shall derive from gml:DefinitionType

All definitions in the application schema shall be declared as global elements in the schema, i.e. they shall be immediate child elements of the XML Schema `<schema>` element. The content model for such global elements shall derive either directly or indirectly from `gml:DefinitionType`.

#### **23.10.4 Dictionaries shall derive from `gml:DictionaryType`**

A dictionary in the application schema shall be declared as a global element in the schema, that is it shall be a child element of the XML Schema `<schema>` element. The content model for such global elements shall derive either directly or indirectly from `gml:DictionaryType`.

### **23.11 Schemas defining Values**

GML allows for user defined value types. Such values types can be used to express the property types of features and other types of GML objects. The basic root types for user-defined values are defined in `basicTypes.xsd`. An alternative form for the expression of values is contained in `valueObjects.xsd`. This is used mainly to provide values for the `gml:resultOf` parameter for an observation.

#### **23.11.1 Target Namespace**

The application schema shall declare a target namespace as described in section 22.3.1 Target Namespace, in which the terms (values) of the vocabulary “live”.

#### **23.11.2 Import `valueObjects.xsd` or `basicTypes.xsd`**

The application schema shall import the necessary components from GML 3 as described in section 22.3.2 Import GML Schema, for example as follows:

```
<import namespace="http://www.opengis.net/gml" schemaLocation="../valueObjects.xsd"/>
```

#### **23.11.3 Construction of New Value Types**

New value types can be created by derivation (typically by restriction) from any of the root types shown in Table 9.

**Table 9**

Content Model	Description
MeasureType	A numerical quantity with a unit of measure (UOM)
CategoryType	A classification
CountType	A count of occurrences, incidences etc.

Some standard value types can be found in the `measures.xsd` schema.

## Annex A (normative)

### Abstract Test Suite

#### A.1 Conformance Class A. Conformance of the XML Implementation of GML data

##### A.1.1 Existence of an applicable GML application schema.

- a) Test Purpose: To verify the existence of a GML application schema applicable to the GML data set.
- b) Test Method: Ensure that the application schema applicable to a particular GML data set exists.
- c) Reference:
- d) Test Type: Basic Test

##### A.1.2 Conformance of the applicable application schema

- a) Test Purpose: To verify that the applicable GML application schema is conformant to this specification.
- b) Test Method: Test whether the application schema satisfies all the tests specified in the Conformance Class B.
- c) Reference:
- d) Test Type: Capability Test.

##### A.1.3 Conformance of the data set

- a) Test Purpose: To verify the validity of the GML data against the conformant GML application schema.
- b) Test Method: Validate GML Data set against corresponding GML application schema tested in the steps A1.1-A1.2. The process may be using an appropriate software tool for validation or be a manual process that checks all relevant definitions from the GML and application schemas.
- c) Reference:
- d) Test Type: Capability Test

#### A.2 Conformance Class B. Conformance of the XML Implementation of GML Application Schema

##### A.2.1 XMLSchema valid

- a) Test Purpose: To verify the conformance of the GML application schema to the XML and XMLSchema specifications.

- b) Test Method: Validate GML application schema using appropriate process for validation. The process may be using an appropriate software tool for validation or be a manual process that checks all relevant rules from the XML 1.0 and XMLSchema 1.0 specifications.
- c) Reference:
- d) Test Type: Capability Test

### **A.2.2 GML model valid**

- a) Test Purpose: To verify the validity against GML model.
- b) Test Method: Ensure that the schema conforms to Clause 23 of this specification (Rules for creating GML Application Schemas) and 22(Profiles).
- c) Reference:
- d) Test Type: Basic Test

## **A.3 Conformance Class C. Conformance of the GML Interface Implementation for GML Data**

### **A.3.1 Serialization capability test**

- a) Test Purpose: To verify the existence of the serialization operation of the implementation.
- b) Test Method: Verify that the implementation implements the serialize operation that produces GML in XML format.
- c) Reference:
- d) Test Type: Basic Test

### **A.3.2 Serialization validity test**

- a) Test Purpose: To verify that the result of the implementation serialization is conformant with this specification.
- b) Test Method: Verify that the result of the serialization of the implementation conforms to this specification according to the Conformance Class A (A.1)
- c) Reference:
- d) Test Type: Capability Test

### **A.3.3 Transforming to and from XML format**

- a) Test Purpose: To verify that the creation of implementation's objects from GML source in XML format, if that capability exists, performs valid function.
- b) Test Method: If it is possible to create implementation's objects from GML source in XML format, verify that successive actions of object creation and serialization produce (test A.3.2) the result that is identical to the source (no loss of information).
- c) Reference:

- d) Test Type: Capability Test

## A.4 Conformance Class D. Conformance of the GML Interface Implementation for GML Application Schema

### A.4.1 Serialization capability test

- a) Test Purpose: To verify the existence of the serialization operation of the implementation.
- b) Test Method: Verify that the implementation implements the serialize operation that produces GML Application Schema in XML Schema format.
- c) Reference:
- d) Test Type: Basic Test

### A.4.2 Serialization validity test

- a) Test Purpose: To verify that the result of the implementation serialization is conformant with this specification.
- b) Test Method: Verify that the result of the serialization of the implementation conforms to this specification according to the Conformance Class B (A.2)
- c) Reference:
- d) Test Type: Capability Test

### A.4.3 Transforming to and from XML format

- a) Test Purpose: To verify that the creation of implementation schema objects from GML Application Schema source in XML format, if that capability exists, performs valid function.
- b) Test Method: If it is possible to create implementation's schema objects from GML Application Schema source in XML format, verify that successful actions of object creation and serialization produce (test A.4.2) the result that is identical to the source (without loss of information).
- c) Reference:
- d) Test Type: Capability Test

## Annex B (normative)

### Conformance

#### B.1 GML Object Rules

This clause summarizes the rules for the detection of GML objects. These rules specify the necessary and sufficient conditions for an element.

##### B.1.1 GML Objects

An element  $x$  is a GML object (of some kind) iff its content model derives ultimately from `gml:AbstractGMLType`, so that it may have identity provided by a `gml:id` attribute.

An element  $x$  is a more specific kind of GML object iff its content model derives ultimately from a more specific GML complex type derived from `AbstractGMLType`. For example, an element  $x$  is a GML feature iff its content model derives ultimately from `gml:AbstractFeatureType`. An element  $x$  is a GML geometry iff its content model derives ultimately from `gml:AbstractGeometryType`. And so on.

##### B.1.2 GML Properties

An element  $p$  is a property of a GML object  $x$  iff  $p$  is a child of  $x$  in the XML sense.

#### B.2 Application Schemas

Writers of application schemas shall conform to the rules stated in clauses 7 - 23.

#### B.3 Conformance Classes

##### B.3.1 Conformance Requirements

Conformance with this specification shall be checked using all the relevant tests specified in Annex A. The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance, are specified in ISO 19105: Geographic information — Conformance and Testing.

This OpenGIS® specification defines four conformance classes: A, B, C and D. Any product claiming conformance with one of these classes shall pass all the requirements described in the corresponding abstract test suite specified in the Annex A.

1. **Class A: Conformance of the XML Implementation of GML Data Set.**  
This class applies to GML data encoded using XML.
2. **Class B: Conformance of the XML Implementation of GML Application Schemas.**  
This class applies to GML application schemas encoded using XML Schema schema description language.

**3. Class C: Conformance of Interface Implementations for GML Data.**  
This class applies to implementations of software interfaces that consume and/or produce GML data.

**4. Class D: Conformance of Interface Implementations for GML Application Schemas.**  
This class applies to implementations of software interfaces that consume and/or produce GML application schemas.

## Annex C (normative)

### GML Schemas

#### C.1 basicTypes.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns:gml="http://www.opengis.net/gml"
  xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:sch="http://www.ascc.net/xml/schematron"
  xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:basicTypes:v3.1.0">basicTypes.xsd</appinfo>
    <documentation>
      Generic simpleContent components for use in GML
    </documentation>
    </annotation>
    <!-- ===== -->
    <simpleType name="NullEnumeration">
      <annotation>
        <documentation> Some common reasons for a null value:
          innapplicable - the object does not have a value
          missing - The correct value is not readily available to the sender of this data.
            Furthermore, a correct value may not exist.
          template - the value will be available later
          unknown - The correct value is not known to, and not computable by, the sender of this data.
            However, a correct value probably exists.
          withheld - the value is not divulged
          other:reason - as indicated by "reason" string
          Specific communities may agree to assign more strict semantics when these terms are used in a particular context.
        </documentation>
      </annotation>
      <union>
        <simpleType>
          <restriction base="string">
            <enumeration value="inapplicable"/>
            <enumeration value="missing"/>
            <enumeration value="template"/>
            <enumeration value="unknown"/>
            <enumeration value="withheld"/>
          </restriction>
        </simpleType>
        <simpleType>
          <restriction base="string">
            <pattern value="other:\w{2,}"/>
          </restriction>
        </simpleType>
      </union>
    </simpleType>
    <!-- ===== -->
    <simpleType name="NullType">
      <annotation>
        <documentation>Utility type for null elements. The value may be selected from one of the enumerated tokens, or
          may be a URI in which case this should identify a resource which describes the reason for the null. </documentation>
      </annotation>
      <union memberTypes="gml:NullEnumeration anyURI"/>
    </simpleType>
    <!-- ===== -->
```

```

<element name="Null" type="gml:NullType"/>
<!-- ===== -->
<simpleType name="SignType">
  <annotation>
    <documentation>Utility type used in various places
- e.g. to indicate the direction of topological objects;
 "+" for forwards, or "-" for backwards.</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="-"/>
    <enumeration value="+"/>
  </restriction>
</simpleType>
<!-- ===== -->
<simpleType name="booleanOrNull">
  <annotation>
    <documentation>Union of the XML Schema boolean type and the GML Nulltype. An element which uses this type
may have content which is either a boolean {0,1,true,false} or a value from Nulltype</documentation>
  </annotation>
  <union memberTypes="gml:NullEnumeration boolean anyURI"/>
</simpleType>
<!-- ===== -->
<simpleType name="booleanOrNullList">
  <annotation>
    <documentation>XML List based on the union type defined above. An element declared with this type contains a
space-separated list of boolean values {0,1,true,false} with null values interspersed as needed</documentation>
  </annotation>
  <list itemType="gml:booleanOrNull"/>
</simpleType>
<!-- ===== -->
<simpleType name="booleanList">
  <annotation>
    <documentation>XML List based on XML Schema boolean type. An element of this type contains a space-separated
list of boolean values {0,1,true,false}</documentation>
  </annotation>
  <list itemType="boolean"/>
</simpleType>
<!-- ===== -->
<simpleType name="stringOrNull">
  <annotation>
    <documentation>Union of the XML Schema string type and the GML Nulltype. An element which uses this type may
have content which is either a string or a value from Nulltype. Note that a "string" may contain whitespace. </documentation>
  </annotation>
  <union memberTypes="gml:NullEnumeration string anyURI"/>
</simpleType>
<!-- ===== -->
<simpleType name="NameOrNull">
  <annotation>
    <documentation>Union of the XML Schema Name type and the GML Nulltype. An element which uses this type may
have content which is either a Name or a value from Nulltype. Note that a "Name" may not contain whitespace.
</documentation>
  </annotation>
  <union memberTypes="gml:NullEnumeration Name anyURI"/>
</simpleType>
<!-- ===== -->
<simpleType name="NameOrNullList">
  <annotation>
    <documentation>XML List based on the union type defined above. An element declared with this type contains a
space-separated list of Name values with null values interspersed as needed</documentation>
  </annotation>
  <list itemType="gml:NameOrNull"/>
</simpleType>
<!-- ===== -->
<simpleType name="NameList">
  <annotation>
    <documentation>XML List based on XML Schema Name type. An element of this type contains a space-separated
list of Name values</documentation>
  </annotation>
  <list itemType="Name"/>
</simpleType>
<!-- ===== -->

```

```

<simpleType name="doubleOrNotNull">
  <annotation>
    <documentation>Union of the XML Schema double type and the GML Nulltype. An element which uses this type may have content which is either a double or a value from Nulltype</documentation>
  </annotation>
  <union memberTypes="gml:NullEnumeration double anyURI"/>
</simpleType>
<!-- ===== -->
<simpleType name="doubleOrNotNullList">
  <annotation>
    <documentation>XML List based on the union type defined above. An element declared with this type contains a space-separated list of double values with null values interspersed as needed</documentation>
  </annotation>
  <list itemType="gml:doubleOrNotNull"/>
</simpleType>
<!-- ===== -->
<simpleType name="doubleList">
  <annotation>
    <documentation>XML List based on XML Schema double type. An element of this type contains a space-separated list of double values</documentation>
  </annotation>
  <list itemType="double"/>
</simpleType>
<!-- ===== -->
<simpleType name="integerOrNotNull">
  <annotation>
    <documentation>Union of the XML Schema integer type and the GML Nulltype. An element which uses this type may have content which is either an integer or a value from Nulltype</documentation>
  </annotation>
  <union memberTypes="gml:NullEnumeration integer anyURI"/>
</simpleType>
<!-- ===== -->
<simpleType name="integerOrNotNullList">
  <annotation>
    <documentation>XML List based on the union type defined above. An element declared with this type contains a space-separated list of integer values with null values interspersed as needed</documentation>
  </annotation>
  <list itemType="gml:integerOrNotNull"/>
</simpleType>
<!-- ===== -->
<simpleType name="integerList">
  <annotation>
    <documentation>XML List based on XML Schema integer type. An element of this type contains a space-separated list of integer values</documentation>
  </annotation>
  <list itemType="integer"/>
</simpleType>
<!-- ===== -->
<complexType name="CodeType">
  <annotation>
    <documentation>Name or code with an (optional) authority. Text token. If the codeSpace attribute is present, then its value should identify a dictionary, thesaurus or authority for the term, such as the organisation who assigned the value, or the dictionary from which it is taken. A text string with an optional codeSpace attribute.</documentation>
  </annotation>
  <simpleContent>
    <extension base="string">
      <attribute name="codeSpace" type="anyURI" use="optional"/>
    </extension>
  </simpleContent>
</complexType>
<!-- ===== -->
<complexType name="CodeListType">
  <annotation>
    <documentation>List of values on a uniform nominal scale. List of text tokens. In a list context a token should not include any spaces, so xsd:Name is used instead of xsd:string. If a codeSpace attribute is present, then its value is a reference to a Reference System for the value, a dictionary or code list.</documentation>
  </annotation>

```

```

</annotation>
<simpleContent>
  <extension base="gml:NameList">
    <attribute name="codeSpace" type="anyURI" use="optional"/>
  </extension>
</simpleContent>
</complexType>
<!-- ===== -->
<complexType name="CodeOrNullListType">
  <annotation>
    <documentation>List of values on a uniform nominal scale. List of text tokens.  
In a list context a token should not include any spaces, so xsd:Name is used instead of xsd:string.  
A member of the list may be a typed null.  
If a codeSpace attribute is present, then its value is a reference to  
a Reference System for the value, a dictionary or code list.</documentation>
  </annotation>
  <simpleContent>
    <extension base="gml:NameOrNullList">
      <attribute name="codeSpace" type="anyURI" use="optional"/>
    </extension>
  </simpleContent>
</complexType>
<!-- ===== -->
<complexType name="MeasureType">
  <annotation>
    <documentation>Number with a scale.  
The value of uom (Units Of Measure) attribute is a reference to a Reference System for the amount, either a ratio or position  
scale.</documentation>
  </annotation>
  <simpleContent>
    <extension base="double">
      <attribute name="uom" type="anyURI" use="required"/>
    </extension>
  </simpleContent>
</complexType>
<!-- ===== -->
<complexType name="MeasureListType">
  <annotation>
    <documentation>List of numbers with a uniform scale.  
The value of uom (Units Of Measure) attribute is a reference to  
a Reference System for the amount, either a ratio or position scale.</documentation>
  </annotation>
  <simpleContent>
    <extension base="gml:doubleList">
      <attribute name="uom" type="anyURI" use="required"/>
    </extension>
  </simpleContent>
</complexType>
<!-- ===== -->
<complexType name="MeasureOrNullListType">
  <annotation>
    <documentation>List of numbers with a uniform scale.  
A member of the list may be a typed null.  
The value of uom (Units Of Measure) attribute is a reference to  
a Reference System for the amount, either a ratio or position scale.</documentation>
  </annotation>
  <simpleContent>
    <extension base="gml:doubleOrNullList">
      <attribute name="uom" type="anyURI" use="required"/>
    </extension>
  </simpleContent>
</complexType>
<!-- ===== -->
<complexType name="CoordinatesType">
  <annotation>
    <documentation>Tables or arrays of tuples.  
May be used for text-encoding of values from a table.  
Actually just a string, but allows the user to indicate which characters are used as separators.  
The value of the 'cs' attribute is the separator for coordinate values,  
and the value of the 'ts' attribute gives the tuple separator (a single space by default);  
the default values may be changed to reflect local usage.</documentation>
  </annotation>

```

Defaults to CSV within a tuple, space between tuples.  
 However, any string content will be schema-valid. </documentation>

```

</annotation>
<simpleContent>
  <extension base="string">
    <attribute name="decimal" type="string" default="."/>
    <attribute name="cs" type="string" default=" "/>
    <attribute name="ts" type="string" default="&#x20;"/>
  </extension>
</simpleContent>
</complexType>
<!-- ===== -->
<simpleType name="NCNameList">
  <annotation>
    <documentation>A set of values, representing a list of token with the lexical value space of NCName. The tokens are separated by whitespace.</documentation>
  </annotation>
  <list itemType="NCName"/>
</simpleType>
<!-- ===== -->
<simpleType name="QNameList">
  <annotation>
    <documentation>A set of values, representing a list of token with the lexical value space of QName. The tokens are separated by whitespace.</documentation>
  </annotation>
  <list itemType="QName"/>
</simpleType>
<!-- ===== -->
</schema>

```

## C.2 coordinateOperations.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" version="3.1.0" xml:lang="en">
  <annotation>
    <cappinfo source="urn:opengis:specification:gml:schema-coordinateOperations"/>
    <documentation>
      <name>coordinateOperations.xsd</name>
      <version>3.1.0</version>
      <scope>How to encode coordinate operation definitions. </scope>
      <description>Builds on referenceSystems.xsd to encode the data needed to define coordinate operations, including Transformations, Conversions, and other specific subtypes of operations. Primary editor: Arliss Whiteside. Last updated 2003/10/16. </description>
      <copyright>Copyright (c) 2002-2003 OpenGIS, All Rights Reserved.</copyright>
      <conformance>This schema encodes the Coordinate Operation (CC_) package of the extended UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex C of Topic 2. </conformance>
    </documentation>
  </annotation>
  <!-- ===== -->
  includes and imports
  =====>
  <include schemaLocation="referenceSystems.xsd"/>
  <include schemaLocation="dataQuality.xsd"/>
  <!-- ===== -->
  elements and types
  =====>
  <element name="_CoordinateOperation" type="gml:AbstractCoordinateOperationType" abstract="true"
    substitutionGroup="gml:Definition"/>
  <!-- ===== -->
  <complexType name="AbstractCoordinateOperationBaseType" abstract="true">
    <annotation>
      <documentation>Basic encoding for coordinate operation objects, simplifying and restricting the DefinitionType as needed. </documentation>
    </annotation>
    <complexContent>
      <restriction base="gml:DefinitionType">

```

```

<sequence>
  <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
  <element ref="gml:remarks" minOccurs="0">
    <annotation>
      <documentation>Comments on or information about this coordinate operation, including source
information. </documentation>
    </annotation>
  </element>
  <element ref="gml:coordinateOperationName"/>
</sequence>
<attribute ref="gml:id" use="required"/>
</restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="coordinateOperationName" type="gml:SimpleNameType" substitutionGroup="gml:name">
  <annotation>
    <documentation>The name by which this coordinate operation is identified. </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractCoordinateOperationType" abstract="true">
  <annotation>
    <documentation>A mathematical operation on coordinates that transforms or converts coordinates to another
coordinate reference system. Many but not all coordinate operations (from CRS A to CRS B) also uniquely define the inverse
operation (from CRS B to CRS A). In some cases, the operation method algorithm for the inverse operation is the same as for the
forward algorithm, but the signs of some operation parameter values must be reversed. In other cases, different algorithms are
required for the forward and inverse operations, but the same operation parameter values are used. If (some) entirely different
parameter values are needed, a different coordinate operation shall be defined.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateOperationBaseType">
      <sequence>
        <element ref="gml:coordinateOperationID" minOccurs="0" maxOccurs="unbounded">
          <annotation>
            <documentation>Set of alternative identifications of this coordinate operation. The first
coordinateOperationID, if any, is normally the primary identification code, and any others are aliases. </documentation>
          </annotation>
        </element>
        <element ref="gml:operationVersion" minOccurs="0"/>
        <element ref="gml:validArea" minOccurs="0"/>
        <element ref="gml:scope" minOccurs="0"/>
        <element ref="gml:_positionalAccuracy" minOccurs="0" maxOccurs="unbounded">
          <annotation>
            <documentation>Unordered set of estimates of the impact of this coordinate operation on point
position accuracy. Gives position error estimates for target coordinates of this coordinate operation, assuming no errors in source
coordinates. </documentation>
          </annotation>
        </element>
        <element ref="gml:sourceCRS" minOccurs="0"/>
        <element ref="gml:targetCRS" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="coordinateOperationID" type="gml:IdentifierType">
  <annotation>
    <documentation>An identification of a coordinate operation. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="operationVersion" type="string">
  <annotation>
    <documentation>Version of the coordinate transformation (i.e., instantiation due to the stochastic nature of the
parameters). Mandatory when describing a transformation, and should not be supplied for a conversion. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="sourceCRS" type="gml:CRSRefType">
  <annotation>

```

```

        <documentation>Association to the source CRS (coordinate reference system) of this coordinate operation.
</documentation>
        </annotation>
    </element>
    <!-- ===== -->
<element name="targetCRS" type="gml:CRSRefType">
    <annotation>
        <documentation>Association to the target CRS (coordinate reference system) of this coordinate operation. For constraints on multiplicity of "sourceCRS" and "targetCRS", see UML model of Coordinate Operation package in OGC Abstract Specification topic 2. </documentation>
    </annotation>
</element>
    <!-- ===== -->
<element name="coordinateOperationRef" type="gml:CoordinateOperationRefType"
substitutionGroup="gml:dictionaryEntry">
    <!-- ===== -->
<complexType name="CoordinateOperationRefType">
    <annotation>
        <documentation>Association to a coordinate operation, either referencing or containing the definition of that coordinate operation. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:DictionaryEntryType">
            <sequence>
                <element ref="gml:_CoordinateOperation" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
    <!-- ===== -->
<element name="ConcatenatedOperation" type="gml:ConcatenatedOperationType"
substitutionGroup="gml:_CoordinateOperation">
    <!-- ===== -->
<complexType name="ConcatenatedOperationType">
    <annotation>
        <documentation>An ordered sequence of two or more single coordinate operations. The sequence of operations is constrained by the requirement that the source coordinate reference system of step (n+1) must be the same as the target coordinate reference system of step (n). The source coordinate reference system of the first step and the target coordinate reference system of the last step are the source and target coordinate reference system associated with the concatenated operation. Instead of a forward operation, an inverse operation may be used for one or more of the operation steps mentioned above, if the inverse operation is uniquely defined by the forward operation.</documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractCoordinateOperationType">
            <sequence>
                <element ref="gml:usesSingleOperation" minOccurs="2" maxOccurs="unbounded">
                    <annotation>
                        <documentation>Ordered sequence of associations to the two or more single operations used by this concatenated operation. </documentation>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
    <!-- ===== -->
<element name="usesSingleOperation" type="gml:SingleOperationRefType">
    <annotation>
        <documentation>Association to a single operation. </documentation>
    </annotation>
</element>
    <!-- ===== -->
<element name="concatenatedOperationRef" type="gml:ConcatenatedOperationRefType"
substitutionGroup="gml:coordinateOperationRef"/>
    <!-- ===== -->
<complexType name="ConcatenatedOperationRefType">
    <annotation>

```

```

<documentation>Association to a concatenated operation, either referencing or containing the definition of that concatenated operation. </documentation>
</annotation>
<complexContent>
  <restriction base="gml:CoordinateOperationRefType">
    <sequence>
      <element ref="gml:ConcatenatedOperation" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="_SingleOperation" type="gml:AbstractSingleOperationType" abstract="true"
substitutionGroup="gml:_CoordinateOperation"/>
<!-- ===== -->
<complexType name="AbstractSingleOperationType" abstract="true">
  <annotation>
    <documentation>A single (not concatenated) coordinate operation. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateOperationType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="singleOperationRef" type="gml:SingleOperationRefType"
substitutionGroup="gml:coordinateOperationRef"/>
<!-- ===== -->
<complexType name="SingleOperationRefType">
  <annotation>
    <documentation>Association to a single operation, either referencing or containing the definition of that single operation. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateOperationRefType">
      <sequence>
        <element ref="gml:_SingleOperation" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="PassThroughOperation" type="gml:PassThroughOperationType"
substitutionGroup="gml:_SingleOperation"/>
<!-- ===== -->
<complexType name="PassThroughOperationType">
  <annotation>
    <documentation>A pass-through operation specifies that a subset of a coordinate tuple is subject to a specific coordinate operation. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractSingleOperationType">
      <sequence>
        <element ref="gml:modifiedCoordinate" maxOccurs="unbounded">
          <annotation>
            <documentation>Ordered sequence of positive integers defining the positions in a coordinate tuple of the coordinates affected by this pass-through operation. </documentation>
          </annotation>
        </element>
        <element ref="gml:usesOperation"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="modifiedCoordinate" type="positiveInteger">
  <annotation>
    <documentation>A positive integer defining a position in a coordinate tuple. </documentation>
  </annotation>
</element>

```

```

<!-- ===== -->
<element name="usesOperation" type="gml:OperationRefType">
  <annotation>
    <documentation>Association to the operation applied to the specified ordinates. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="passThroughOperationRef" type="gml:PassThroughOperationRefType"
substitutionGroup="gml:singleOperationRef"/>
<!-- ===== -->
<complexType name="PassThroughOperationRefType">
  <annotation>
    <documentation>Association to a pass through operation, either referencing or containing the definition of that pass
through operation. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:SingleOperationRefType">
      <sequence>
        <element ref="gml:PassThroughOperation" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="_Operation" type="gml:AbstractOperationType" abstract="true" substitutionGroup="gml:_SingleOperation"/>
<!-- ===== -->
<complexType name="AbstractOperationType" abstract="true">
  <annotation>
    <documentation>A parameterized mathematical operation on coordinates that transforms or converts coordinates to
another coordinate reference system. This coordinate operation uses an operation method, usually with associated parameter
values. However, operation methods and parameter values are directly associated with concrete subtypes, not with this abstract
type. </documentation>
  </annotation>

```

This abstract complexType shall not be directly used, extended, or restricted in a compliant Application Schema.

```

</documentation>
  <complexContent>
    <extension base="gml:AbstractSingleOperationType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="operationRef" type="gml:OperationRefType" substitutionGroup="gml:singleOperationRef"/>
<!-- ===== -->
<complexType name="OperationRefType">
  <annotation>
    <documentation>Association to an abstract operation, either referencing or containing the definition of that operation. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:SingleOperationRefType">
      <sequence>
        <element ref="gml:_Operation" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="_GeneralConversion" type="gml:AbstractGeneralConversionType" abstract="true"
substitutionGroup="gml:_Operation"/>
<!-- ===== -->
<complexType name="AbstractGeneralConversionType" abstract="true">
  <annotation>
    <documentation>An abstract operation on coordinates that does not include any change of datum. The best-known
example of a coordinate conversion is a map projection. The parameters describing coordinate conversions are defined rather
than empirically derived. Note that some conversions have no parameters. </documentation>
  </annotation>

```

This abstract complexType is expected to be extended for well-known operation methods with many Conversion instances, in Application Schemas that define operation-method-specialized element names and contents. This conversion uses an operation method, usually with associated parameter values. However, operation methods and parameter values are directly associated with concrete subtypes, not with this abstract type. All concrete types derived from this type shall extend this type to include a "usesMethod" element that references the "OperationMethod" element. Similarly, all concrete types derived from this type shall extend this type to include zero or more elements each named "uses...Value" that each use the type of an element substitutable for the "\_generalParameterValue" element. *</documentation>*

```

</annotation>
<complexContent>
  <restriction base="gml:AbstractOperationType">
    <sequence>
      <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:remarks" minOccurs="0"/>
      <element ref="gml:coordinateOperationName"/>
      <element ref="gml:coordinateOperationID" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:validArea" minOccurs="0"/>
      <element ref="gml:scope" minOccurs="0"/>
      <element ref="gml:_positionalAccuracy" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
    <attribute ref="gml:id" use="required"/>
  </restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="generalConversionRef" type="gml:GeneralConversionRefType" substitutionGroup="gml:operationRef"/>
<!-- ===== -->
<complexType name="GeneralConversionRefType">
  <annotation>
    <documentation>Association to a general conversion, either referencing or containing the definition of that conversion. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:OperationRefType">
      <sequence>
        <element ref="gml:_GeneralConversion" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="Conversion" type="gml:ConversionType" substitutionGroup="gml:_GeneralConversion"/>
<!-- ===== -->
<complexType name="ConversionType">
  <annotation>
    <documentation>A concrete operation on coordinates that does not include any change of Datum. The best-known example of a coordinate conversion is a map projection. The parameters describing coordinate conversions are defined rather than empirically derived. Note that some conversions have no parameters. </documentation>
  </annotation>
</complexType>

```

This concrete complexType can be used with all operation methods, without using an Application Schema that defines operation-method-specialized element names and contents, especially for methods with only one Conversion instance. *</documentation>*

```

</annotation>
<complexContent>
  <extension base="gml:AbstractGeneralConversionType">
    <sequence>
      <element ref="gml:usesMethod"/>
      <element ref="gml:usesValue" minOccurs="0" maxOccurs="unbounded">
        <annotation>
          <documentation>Unordered list of composition associations to the set of parameter values used by this conversion operation. </documentation>
        </annotation>
        <elements>
          <element name="usesValue" type="gml:GeneralParameterValueType"/>
        </elements>
      </element>
    </sequence>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="usesMethod" type="gml:OperationMethodRefType">
  <annotation>
    <documentation>Association to the operation method used by this coordinate operation. </documentation>
  </annotation>
</element>

```

```

</element>
<!-- ===== -->
<element name="usesValue" type="gml:ParameterValueType">
  <annotation>
    <documentation>Composition association to a parameter value used by this coordinate operation. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="conversionRef" type="gml:ConversionRefType" substitutionGroup="gml:generalConversionRef"/>
<!-- ===== -->
<complexType name="ConversionRefType">
  <annotation>
    <documentation>Association to a concrete general-purpose conversion, either referencing or containing the definition of that conversion. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:GeneralConversionRefType">
      <sequence>
        <element ref="gml:Conversion" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="_GeneralTransformation" type="gml:AbstractGeneralTransformationType" abstract="true" substitutionGroup="gml:_Operation"/>
<!-- ===== -->
<complexType name="AbstractGeneralTransformationType" abstract="true">
  <annotation>
    <documentation>An abstract operation on coordinates that usually includes a change of Datum. The parameters of a coordinate transformation are empirically derived from data containing the coordinates of a series of points in both coordinate reference systems. This computational process is usually "over-determined", allowing derivation of error (or accuracy) estimates for the transformation. Also, the stochastic nature of the parameters may result in multiple (different) versions of the same coordinate transformation. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:AbstractOperationType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0"/>
        <element ref="gml:coordinateOperationName"/>
        <element ref="gml:coordinateOperationID" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:operationVersion"/>
        <element ref="gml:validArea" minOccurs="0"/>
        <element ref="gml:scope" minOccurs="0"/>
        <element ref="gml:_operationalAccuracy" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:sourceCRS"/>
        <element ref="gml:targetCRS"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="generalTransformationRef" type="gml:GeneralTransformationRefType" substitutionGroup="gml:operationRef"/>
<!-- ===== -->
<complexType name="GeneralTransformationRefType">
  <annotation>

```

```

<documentation>Association to a general transformation, either referencing or containing the definition of that
transformation. </documentation>
</annotation>
<complexContent>
  <restriction base="gml:OperationRefType">
    <sequence>
      <element ref="gml:_GeneralTransformation" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="Transformation" type="gml:TransformationType" substitutionGroup="gml:_GeneralTransformation"/>
<!-- ===== -->
<complexType name="TransformationType">
  <annotation>
    <documentation>A concrete operation on coordinates that usually includes a change of datum. The parameters of a
coordinate transformation are empirically derived from data containing the coordinates of a series of points in both coordinate
reference systems. This computational process is usually "over-determined", allowing derivation of error (or accuracy) estimates
for the transformation. Also, the stochastic nature of the parameters may result in multiple (different) versions of the same
coordinate transformation. </documentation>
  </annotation>
</complexType>

```

This concrete complexType can be used for all operation methods, without using an Application Schema that defines operation-method-specialized element names and contents, especially for methods with only one Transformation instance.

```

</documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractGeneralTransformationType">
    <sequence>
      <element ref="gml:usesMethod"/>
      <element ref="gml:usesValue" minOccurs="0" maxOccurs="unbounded">
        <annotation>
          <documentation>Unordered set of composition associations to the set of parameter values used by
this transformation operation. </documentation>
        </annotation>
      </element>
    </sequence>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="transformationRef" type="gml:TransformationRefType"
substitutionGroup="gml:generalTransformationRef"/>
<!-- ===== -->
<complexType name="TransformationRefType">
  <annotation>
    <documentation>Association to a transformation, either referencing or containing the definition of that transformation. </documentation>
  </annotation>
</complexType>
</documentation>
</annotation>
<complexContent>
  <restriction base="gml:GeneralTransformationRefType">
    <sequence>
      <element ref="gml:Transformation" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </restriction>
</complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="_generalParameterValue" type="gml:AbstractGeneralParameterValueType" abstract="true"/>
<!-- ===== -->
<complexType name="AbstractGeneralParameterValueType" abstract="true">
  <annotation>
    <documentation>Abstract parameter value or group of parameter values. </documentation>
  </annotation>

```

This abstract complexType is expected to be extended and restricted for well-known operation methods with many instances, in Application Schemas that define operation-method-specialized element names and contents. Specific parameter value elements are directly contained in concrete subtypes, not in this abstract type. All concrete types derived from this type shall extend this type to include one "...Value" element with an appropriate type, which should be one of the element types allowed in the

ParameterValue. In addition, all derived concrete types shall extend this type to include a "valueOfParameter" element that references one element substitutable for the "OperationParameter" element. *</documentation>*

```
</annotation>
<sequence/>
</complexType>
<!-- ===== -->
<element name="parameterValue" type="gml:ParameterValueType" substitutionGroup="gml:_generalParameterValue"/>
<!-- ===== -->
<complexType name="ParameterValueType">
  <annotation>
    <documentation>A parameter value, ordered sequence of values, or reference to a file of parameter values. This concrete complexType can be used for operation methods without using an Application Schema that defines operation-method-specialized element names and contents, especially for methods with only one instance. This complexType can be used, extended, or restricted for well-known operation methods, especially for methods with many instances. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGeneralParameterValueType">
      <sequence>
        <choice>
          <element ref="gml:value"/>
          <element ref="gml:dmsAngleValue"/>
          <element ref="gml:stringValue"/>
          <element ref="gml:integerValue"/>
          <element ref="gml:booleanValue"/>
          <element ref="gml:valueList"/>
          <element ref="gml:integerValueList"/>
          <element ref="gml:valueFile"/>
        </choice>
        <element ref="gml:valueOfParameter"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="value" type="gml:MeasureType">
  <annotation>
    <documentation>Numeric value of an operation parameter, with its associated unit of measure. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="dmsAngleValue" type="gml:DMSAngleType">
  <annotation>
    <documentation>Value of an angle operation parameter, in either degree-minute-second format or single value format. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="stringValue" type="string">
  <annotation>
    <documentation>String value of an operation parameter. A string value does not have an associated unit of measure. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="integerValue" type="positiveInteger">
  <annotation>
    <documentation>Positive integer value of an operation parameter, usually used for a count. An integer value does not have an associated unit of measure. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="booleanValue" type="boolean">
  <annotation>
    <documentation>Boolean value of an operation parameter. A Boolean value does not have an associated unit of measure. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="valueList" type="gml:MeasureListType">
```

```

<annotation>
  <documentation>Ordered sequence of two or more numeric values of an operation parameter list, where each value
has the same associated unit of measure. An element of this type contains a space-separated sequence of double values.
</documentation>
</annotation>
</element>
<!-- ===== -->
<element name="integerValueList" type="gml:integerList">
  <annotation>
    <documentation>Ordered sequence of two or more integer values of an operation parameter list, usually used for
counts. These integer values do not have an associated unit of measure. An element of this type contains a space-separated
sequence of integer values. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="valueFile" type="anyURI">
  <annotation>
    <documentation>Reference to a file or a part of a file containing one or more parameter values, each numeric value
with its associated unit of measure. When referencing a part of a file, that file must contain multiple identified parts, such as an
XML encoded document. Furthermore, the referenced file or part of a file can reference another part of the same or different files,
as allowed in XML documents. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="valueOfParameter" type="gml:OperationParameterRefType">
  <annotation>
    <documentation>Association to the operation parameter that this is a value of. </documentation>
  </annotation>
</element>
<!-- ===== -->
<!-- ===== -->
<element name="parameterValueGroup" type="gml:ParameterValueGroupType"
substitutionGroup="gml:_generalParameterValue"/>
<!-- ===== -->
<complexType name="ParameterValueGroupType">
  <annotation>
    <documentation>A group of related parameter values. The same group can be repeated more than once in a
Conversion, Transformation, or higher level parameterValueGroup, if those instances contain different values of one or more
parameterValues which suitably distinguish among those groups. This concrete complexType can be used for operation methods
without using an Application Schema that defines operation-method-specialized element names and contents, especially for
methods with only one instance. This complexType can be used, extended, or restricted for well-known operation methods,
especially for methods with many instances. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGeneralParameterValueType">
      <sequence>
        <element ref="gml:includesValue" minOccurs="2" maxOccurs="unbounded">
          <annotation>
            <documentation>Unordered set of composition associations to the parameter values and groups of
values included in this group. </documentation>
          </annotation>
        </element>
        <element ref="gml:valuesOfGroup"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="includesValue" type="gml:AbstractGeneralParameterValueType"
substitutionGroup="gml:_generalParameterValue">
  <annotation>
    <documentation>A composition association to a parameter value or group of values included in this group.
</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="valuesOfGroup" type="gml:OperationParameterGroupRefType">
  <annotation>
    <documentation>Association to the operation parameter group for which this element provides parameter values.
</documentation>
  </annotation>
</element>

```

```

</element>
<!-- ===== -->
<!-- ===== -->
<element name="OperationMethod" type="gml:OperationMethodType" substitutionGroup="gml:Definition"/>
<!-- ===== -->
<!-- ===== -->
<complexType name="OperationMethodBaseType" abstract="true">
    <annotation>
        <documentation>Basic encoding for operation method objects, simplifying and restricting the DefinitionType as needed. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:DefinitionType">
            <sequence>
                <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:remarks" minOccurs="0">
                    <annotation>
                        <documentation>Comments on or information about this operation method, including source information.</documentation>
                    </annotation>
                </element>
                <element ref="gml:methodName"/>
            </sequence>
            <attribute ref="gml:id" use="required"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="methodName" type="gml:SimpleNameType" substitutionGroup="gml:name">
    <annotation>
        <documentation>The name by which this operation method is identified. </documentation>
    </annotation>
</element>
<!-- ===== -->
<!-- ===== -->
<complexType name="OperationMethodType">
    <annotation>
        <documentation>Definition of an algorithm used to perform a coordinate operation. Most operation methods use a number of operation parameters, although some coordinate conversions use none. Each coordinate operation using the method assigns values to these parameters. </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:OperationMethodBaseType">
            <sequence>
                <element ref="gml:methodID" minOccurs="0" maxOccurs="unbounded">
                    <annotation>
                        <documentation>Set of alternative identifications of this operation method. The first methodID, if any, is normally the primary identification code, and any others are aliases. </documentation>
                    </annotation>
                </element>
                <element ref="gml:methodFormula"/>
                <element ref="gml:sourceDimensions"/>
                <element ref="gml:targetDimensions"/>
                <element ref="gml:usesParameter" minOccurs="0" maxOccurs="unbounded">
                    <annotation>
                        <documentation>Unordered list of associations to the set of operation parameters and parameter groups used by this operation method. </documentation>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="methodID" type="gml:IdentifierType">
    <annotation>
        <documentation>An identification of an operation method. </documentation>
    </annotation>
</element>
<!-- ===== -->
<!-- ===== -->
<element name="methodFormula" type="gml:CodeType">

```

```

<annotation>
  <documentation>Formula(s) used by this operation method. The value may be a reference to a publication. Note that
the operation method may not be analytic, in which case this element references or contains the procedure, not an analytic
formula.</documentation>
</annotation>
</element>
<!-- ===== -->
<element name="sourceDimensions" type="positiveInteger">
  <annotation>
    <documentation>Number of dimensions in the source CRS of this operation method. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="targetDimensions" type="positiveInteger">
  <annotation>
    <documentation>Number of dimensions in the target CRS of this operation method. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="usesParameter" type="gml:AbstractGeneralOperationParameterRefType">
  <annotation>
    <documentation>Association to an operation parameter or parameter group used by this operation method.
</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="operationMethodRef" type="gml:OperationMethodRefType" substitutionGroup="gml:dictionaryEntry"/>
<!-- ===== -->
<complexType name="OperationMethodRefType">
  <annotation>
    <documentation>Association to a concrete general-purpose operation method, either referencing or containing the
definition of that method. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:OperationMethod" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="_GeneralOperationParameter" type="gml:AbstractGeneralOperationParameterType" abstract="true"
substitutionGroup="gml:Definition"/>
<!-- ===== -->
<complexType name="AbstractGeneralOperationParameterType" abstract="true">
  <annotation>
    <documentation>Abstract definition of a parameter or group of parameters used by an operation method.
</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:DefinitionType">
      <sequence>
        <element ref="gml:minimumOccurs" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="minimumOccurs" type="nonNegativeInteger">
  <annotation>
    <documentation>The minimum number of times that values for this parameter group or parameter are required. If this
attribute is omitted, the minimum number is one. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="abstractGeneralOperationParameterRef" type="gml:AbstractGeneralOperationParameterRefType"
substitutionGroup="gml:dictionaryEntry"/>
<!-- ===== -->

```

```

<complexType name="AbstractGeneralOperationParameterRefType">
  <annotation>
    <documentation>Association to an operation parameter or group, either referencing or containing the definition of that parameter or group. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:_GeneralOperationParameter" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="OperationParameter" type="gml:OperationParameterType"
substitutionGroup="gml:_GeneralOperationParameter"/>
<!-- ===== -->
<complexType name="OperationParameterBaseType" abstract="true">
  <annotation>
    <documentation>Basic encoding for operation parameter objects, simplifying and restricting the DefinitionType as needed. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:AbstractGeneralOperationParameterType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0">
          <annotation>
            <documentation>Comments on or information about this operation parameter, including source information. </documentation>
          </annotation>
        </element>
        <element ref="gml:parameterName"/>
        <element ref="gml:minimumOccurs" minOccurs="0"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="parameterName" type="gml:SimpleNameType" substitutionGroup="gml:name">
  <annotation>
    <documentation>The name by which this operation parameter is identified. </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="OperationParameterType">
  <annotation>
    <documentation>The definition of a parameter used by an operation method. Most parameter values are numeric, but other types of parameter values are possible. This complexType is expected to be used or extended for all operation methods, without defining operation-method-specialized element names. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:OperationParameterBaseType">
      <sequence>
        <element ref="gml:parameterID" minOccurs="0" maxOccurs="unbounded">
          <annotation>
            <documentation>Set of alternative identifications of this operation parameter. The first parameterID, if any, is normally the primary identification code, and any others are aliases. </documentation>
          </annotation>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="parameterID" type="gml:IdentifierType">
  <annotation>

```

```

        <documentation>An identification of an operation parameter. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="operationParameterRef" type="gml:OperationParameterRefType"/>
<!-- ===== -->
<complexType name="OperationParameterRefType">
    <annotation>
        <documentation>Association to an operation parameter, either referencing or containing the definition of that
parameter. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:AbstractGeneralOperationParameterRefType">
            <sequence>
                <element ref="gml:OperationParameter" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="OperationParameterGroup" type="gml:OperationParameterGroupType"
substitutionGroup="gml:_GeneralOperationParameter"/>
<!-- ===== -->
<complexType name="OperationParameterGroupBaseType" abstract="true">
    <annotation>
        <documentation>Basic encoding for operation parameter group objects, simplifying and restricting the DefinitionType
as needed. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:AbstractGeneralOperationParameterType">
            <sequence>
                <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:remarks" minOccurs="0">
                    <annotation>
                        <documentation>Comments on or information about this operation parameter group, including source
information. </documentation>
                    </annotation>
                </element>
                <element ref="gml:groupName"/>
                <element ref="gml:minimumOccurs" minOccurs="0"/>
            </sequence>
            <attribute ref="gml:id" use="required"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="groupName" type="gml:SimpleNameType" substitutionGroup="gml:name">
    <annotation>
        <documentation>The name by which this operation parameter group is identified. </documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="OperationParameterGroupType">
    <annotation>
        <documentation>The definition of a group of parameters used by an operation method. This complexType is
expected to be used or extended for all applicable operation methods, without defining operation-method-specialized element
names. </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:OperationParameterGroupBaseType">
            <sequence>
                <element ref="gml:groupID" minOccurs="0" maxOccurs="unbounded">
                    <annotation>
                        <documentation>Set of alternative identifications of this operation parameter group. The first groupID,
if any, is normally the primary identification code, and any others are aliases. </documentation>
                    </annotation>
                </element>
                <element ref="gml:maximumOccurs" minOccurs="0"/>
                <element ref="gml:includesParameter" minOccurs="2" maxOccurs="unbounded">
                    <annotation>

```

```

        <documentation>Unordered list of associations to the set of operation parameters that are members
of this group. </documentation>
        </annotation>
        </element>
        </sequence>
        </extension>
        </complexContent>
    </complexType>
    <!-- ===== -->
<element name="groupID" type="gml:IdentifierType">
    <annotation>
        <documentation>An identification of an operation parameter group. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="maximumOccurs" type="positiveInteger">
    <annotation>
        <documentation>The maximum number of times that values for this parameter group can be included. If this attribute
is omitted, the maximum number is one. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="includesParameter" type="gml:AbstractGeneralOperationParameterRefType">
    <annotation>
        <documentation>Association to an operation parameter that is a member of a group. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="operationParameterGroupRef" type="gml:OperationParameterRefType"/>
<!-- ===== -->
<complexType name="OperationParameterGroupRefType">
    <annotation>
        <documentation>Association to an operation parameter, either referencing or containing the definition of that
parameter. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:AbstractGeneralOperationParameterRefType">
            <sequence>
                <element ref="gml:OperationParameterGroup" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
</schema>

```

### C.3 coordinateReferenceSystems.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="3.1.0" xml:lang="en">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-coordinateReferenceSystems"/>
        <documentation>
            <name>coordinateReferenceSystems.xsd</name>
            <version>3.1.0</version>
            <scope>How to encode coordinate reference system definitions. </scope>
            <description>Builds on referenceSystems.xsd to encode the data needed to define coordinate reference systems,
including the specific subtypes of coordinate reference systems. Primary editor: Arliss Whiteside. Last updated 2003/10/16.
            </description>
            <copyright>Copyright (c) 2002-2003 Open GIS Consortium, All Rights Reserved.</copyright>
            <conformance>This schema encodes the Coordinate Reference System (SC_) package of the extended UML Model
for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates, with the exception of the abstract "SC_CRS" class.
The "SC_CRS" class is encoded in referenceSystems.xsd, to eliminate the (circular) references from coordinateOperations.xsd to
coordinateReferenceSystems.xsd. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as
described in Annex C of Topic 2. </conformance>
        </documentation>

```

```

</annotation>
<!-- =====
includes and imports
===== -->
<include schemaLocation="coordinateSystems.xsd"/>
<include schemaLocation="datums.xsd"/>
<include schemaLocation="coordinateOperations.xsd"/>
<!-- =====
elements and types
===== -->
<element name="_CoordinateReferenceSystem" type="gml:AbstractCoordinateReferenceSystemType" abstract="true"
substitutionGroup="gml:_CRS"/>
<!-- =====
<complexType name="AbstractCoordinateReferenceSystemType" abstract="true">
<annotation>
<documentation>A coordinate reference system consists of an ordered sequence of coordinate system axes that are
related to the earth through a datum. A coordinate reference system is defined by one datum and by one coordinate system. Most
coordinate reference system do not move relative to the earth, except for engineering coordinate reference systems defined on
moving platforms such as cars, ships, aircraft, and spacecraft. For further information, see OGC Abstract Specification Topic 2.


Coordinate reference systems are commonly divided into sub-types. The common classification criterion for sub-typing of
coordinate reference systems is the way in which they deal with earth curvature. This has a direct effect on the portion of the
earth's surface that can be covered by that type of CRS with an acceptable degree of error. The exception to the rule is the
subtype "Temporal" which has been added by analogy. </documentation>


</annotation>
<complexContent>
<extension base="gml:AbstractCRSType"/>
</complexContent>
</complexType>
<!-- =====
<element name="coordinateReferenceSystemRef" type="gml:CoordinateReferenceSystemRefType"
substitutionGroup="gml:crsRef"/>
===== -->
<complexType name="CoordinateReferenceSystemRefType">
<annotation>
<documentation>Association to a coordinate reference system, either referencing or containing the definition of that
reference system. </documentation>
</annotation>
<complexContent>
<restriction base="gml:CRSRefType">
<sequence>
<element ref="gml:_CoordinateReferenceSystem" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</restriction>
</complexContent>
</complexType>
<!-- =====
<element name="CompoundCRS" type="gml:CompoundCRSType" substitutionGroup="gml:_CRS"/>
===== -->
<complexType name="CompoundCRSType">
<annotation>
<documentation>A coordinate reference system describing the position of points through two or more independent
coordinate reference systems. </documentation>
</annotation>
<complexContent>
<extension base="gml:AbstractCRSType">
<sequence>
<element ref="gml:includesCRS" minOccurs="2" maxOccurs="unbounded">
<annotation>
<documentation>Ordered sequence of associations to all the component coordinate reference
systems included in this compound coordinate reference system. </documentation>
</annotation>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
<!-- =====
<element name="includesCRS" type="gml:CoordinateReferenceSystemRefType">
<annotation>

```

```

<documentation>An association to a component coordinate reference system included in this compound coordinate reference system. </documentation>
</annotation>
</element>
<!-- ===== -->
<element name="compoundCRSRef" type="gml:CompoundCRSRefType" substitutionGroup="gml:crsRef"/>
<!-- ===== -->
<complexType name="CompoundCRSRefType">
  <annotation>
    <documentation>Association to a compound coordinate reference system, either referencing or containing the definition of that reference system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CRSRefType">
      <sequence>
        <element ref="gml:CompoundCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="GeographicCRS" type="gml:GeographicCRSType" substitutionGroup="gml:_CoordinateReferenceSystem"/>
<!-- ===== -->
<complexType name="GeographicCRSType">
  <annotation>
    <documentation>A coordinate reference system based on an ellipsoidal approximation of the geoid; this provides an accurate representation of the geometry of geographic features for a large portion of the earth's surface.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <element ref="gml:usesEllipsoidalCS"/>
        <element ref="gml:usesGeodeticDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="usesEllipsoidalCS" type="gml:EllipsoidalCSRefType">
  <annotation>
    <documentation>Association to the ellipsoidal coordinate system used by this CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="usesGeodeticDatum" type="gml:GeodeticDatumRefType">
  <annotation>
    <documentation>Association to the geodetic datum used by this CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="geographicCRSRef" type="gml:GeographicCRSRefType" substitutionGroup="gml:coordinateReferenceSystemRef"/>
<!-- ===== -->
<complexType name="GeographicCRSRefType">
  <annotation>
    <documentation>Association to a geographic coordinate reference system, either referencing or containing the definition of that reference system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:GeographicCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

```

<!-- ===== -->
<element name="VerticalCRS" type="gml:VerticalCRSType" substitutionGroup="gml:_CoordinateReferenceSystem"/>
<!-- ===== -->
<complexType name="VerticalCRSType">
  <annotation>
    <documentation>A 1D coordinate reference system used for recording heights or depths. Vertical CRSs make use of the direction of gravity to define the concept of height or depth, but the relationship with gravity may not be straightforward. By implication, ellipsoidal heights (h) cannot be captured in a vertical coordinate reference system. Ellipsoidal heights cannot exist independently, but only as an inseparable part of a 3D coordinate tuple defined in a geographic 3D coordinate reference system.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <element ref="gml:usesVerticalCS"/>
        <element ref="gml:usesVerticalDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="usesVerticalCS" type="gml:VerticalCSRefType">
  <annotation>
    <documentation>Association to the vertical coordinate system used by this CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="usesVerticalDatum" type="gml:VerticalDatumRefType">
  <annotation>
    <documentation>Association to the vertical datum used by this CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="verticalCRSRef" type="gml:VerticalCRSRefType"
substitutionGroup="gml:coordinateReferenceSystemRef"/>
<!-- ===== -->
<complexType name="VerticalCRSRefType">
  <annotation>
    <documentation>Association to a vertical coordinate reference system, either referencing or containing the definition of that reference system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:VerticalCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="GeocentricCRS" type="gml:GeocentricCRSType" substitutionGroup="gml:_CoordinateReferenceSystem"/>
<!-- ===== -->
<complexType name="GeocentricCRSType">
  <annotation>
    <documentation>A 3D coordinate reference system with the origin at the approximate centre of mass of the earth. A geocentric CRS deals with the earth's curvature by taking a 3D spatial view, which obviates the need to model the earth's curvature. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <choice>
          <element ref="gml:usesCartesianCS"/>
          <element ref="gml:usesSphericalCS"/>
        </choice>
        <element ref="gml:usesGeodeticDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

<!-- ===== -->
<element name="usesCartesianCS" type="gml:CartesianCSRefType">
  <annotation>
    <documentation>Association to the Cartesian coordinate system used by this CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="usesSphericalCS" type="gml:SphericalCSRefType">
  <annotation>
    <documentation>Association to the spherical coordinate system used by this CRS.</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="geocentricCRSRef" type="gml:GeocentricCRSRefType"
substitutionGroup="gml:coordinateReferenceSystemRef"/>
<!-- ===== -->
<complexType name="GeocentricCRSRefType">
  <annotation>
    <documentation>Association to a geocentric coordinate reference system, either referencing or containing the definition of that reference system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:GeocentricCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="_GeneralDerivedCRS" type="gml:AbstractGeneralDerivedCRSType" abstract="true"
substitutionGroup="gml:_CoordinateReferenceSystem"/>
<!-- ===== -->
<complexType name="AbstractGeneralDerivedCRSType" abstract="true">
  <annotation>
    <documentation>A coordinate reference system that is defined by its coordinate conversion from another coordinate reference system (not by a datum). This abstract complexType shall not be used, extended, or restricted, in an Application Schema, to define a concrete subtype with a meaning equivalent to a concrete subtype specified in this document. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <element ref="gml:baseCRS"/>
        <element ref="gml:definedByConversion"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="baseCRS" type="gml:CoordinateReferenceSystemRefType">
  <annotation>
    <documentation>Association to the coordinate reference system used by this derived CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="definedByConversion" type="gml:GeneralConversionRefType">
  <annotation>
    <documentation>Association to the coordinate conversion used to define this derived CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="generalDerivedCRSRef" type="gml:GeneralDerivedCRSRefType"
substitutionGroup="gml:coordinateReferenceSystemRef"/>
<!-- ===== -->
<complexType name="GeneralDerivedCRSRefType">
  <annotation>

```

```

<documentation>Association to a general derived coordinate reference system, either referencing or containing the
definition of that reference system. </documentation>
</annotation>
<complexContent>
  <restriction base="gml:CoordinateReferenceSystemRefType">
    <sequence>
      <element ref="gml:_GeneralDerivedCRS" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="ProjectedCRS" type="gml:ProjectedCRSType" substitutionGroup="gml:_GeneralDerivedCRS"/>
<!-- ===== -->
<complexType name="ProjectedCRSType">
  <annotation>
    <documentation>A 2D coordinate reference system used to approximate the shape of the earth on a planar surface,
but in such a way that the distortion that is inherent to the approximation is carefully controlled and known. Distortion correction is
commonly applied to calculated bearings and distances to produce values that are a close match to actual field values.
</documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractGeneralDerivedCRSType">
        <sequence>
          <element ref="gml:usesCartesianCS"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
<!-- ===== -->
<element name="projectedCRSRef" type="gml:ProjectedCRSRefType" substitutionGroup="gml:generalDerivedCRSRef"/>
<!-- ===== -->
<complexType name="ProjectedCRSRefType">
  <annotation>
    <documentation>Association to a projected coordinate reference system, either referencing or containing the
definition of that reference system. </documentation>
    </annotation>
    <restriction base="gml:GeneralDerivedCRSRefType">
      <sequence>
        <element ref="gml:ProjectedCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="DerivedCRS" type="gml:DerivedCRSType" substitutionGroup="gml:_GeneralDerivedCRS"/>
<!-- ===== -->
<complexType name="DerivedCRSType">
  <annotation>
    <documentation>A coordinate reference system that is defined by its coordinate conversion from another coordinate
reference system but is not a projected coordinate reference system. This category includes coordinate reference systems
derived from a projected coordinate reference system. </documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractGeneralDerivedCRSType">
        <sequence>
          <element ref="gml:derivedCRSType"/>
          <element ref="gml:usesCS"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
<!-- ===== -->
<element name="derivedCRSType" type="gml:DerivedCRSTypeType"/>
<!-- ===== -->
<complexType name="DerivedCRSTypeType">
  <annotation>
    <documentation>Type of a derived coordinate reference system. </documentation>
    </annotation>
  </complexType>

```

```

</annotation>
<simpleContent>
  <restriction base="gml:CodeType">
    <attribute name="codeSpace" type="anyURI" use="required">
      <annotation>
        <documentation>Reference to a source of information specifying the values and meanings of all the allowed string values for this DerivedCRSTypeType. </documentation>
      </annotation>
    </attribute>
  </restriction>
</simpleContent>
</complexType>
<!-- ===== -->
<element name="usesCS" type="gml:CoordinateSystemRefType">
  <annotation>
    <documentation>Association to the coordinate system used by this CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="derivedCRSRef" type="gml:DerivedCRSRefType" substitutionGroup="gml:generalDerivedCRSRef"/>
<!-- ===== -->
<complexType name="DerivedCRSRefType">
  <annotation>
    <documentation>Association to a non-projected derived coordinate reference system, either referencing or containing the definition of that reference system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:GeneralDerivedCRSRefType">
      <sequence>
        <element ref="gml:DerivedCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="EngineeringCRS" type="gml:EngineeringCRSType" substitutionGroup="gml:_CoordinateReferenceSystem"/>
<!-- ===== -->
<complexType name="EngineeringCRSType">
  <annotation>
    <documentation>A contextually local coordinate reference system; which can be divided into two broad categories:  

- earth-fixed systems applied to engineering activities on or near the surface of the earth;  

- CRSs on moving platforms such as road vehicles, vessels, aircraft, or spacecraft.  

For further information, see OGC Abstract Specification Topic 2. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <element ref="gml:usesCS"/>
        <element ref="gml:usesEngineeringDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="usesEngineeringDatum" type="gml:EngineeringDatumRefType">
  <annotation>
    <documentation>Association to the engineering datum used by this CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="engineeringCRSRef" type="gml:EngineeringCRSRefType" substitutionGroup="gml:coordinateReferenceSystemRef"/>
<!-- ===== -->
<complexType name="EngineeringCRSRefType">
  <annotation>
    <documentation>Association to an engineering coordinate reference system, either referencing or containing the definition of that reference system. </documentation>
  </annotation>

```

```

</annotation>
<complexContent>
  <restriction base="gml:CoordinateReferenceSystemRefType">
    <sequence>
      <element ref="gml:EngineeringCRS" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="ImageCRS" type="gml:ImageCRSType" substitutionGroup="gml:_CoordinateReferenceSystem"/>
<!-- ===== -->
<complexType name="ImageCRSType">
  <annotation>
    <documentation>An engineering coordinate reference system applied to locations in images. Image coordinate reference systems are treated as a separate sub-type because a separate user community exists for images with its own terms of reference. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <choice>
          <element ref="gml:usesCartesianCS"/>
          <element ref="gml:usesObliqueCartesianCS"/>
        </choice>
        <element ref="gml:usesImageDatum"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="usesObliqueCartesianCS" type="gml:ObliqueCartesianCSRefType">
  <annotation>
    <documentation>Association to the oblique Cartesian coordinate system used by this CRS.</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="usesImageDatum" type="gml:ImageDatumRefType">
  <annotation>
    <documentation>Association to the image datum used by this CRS. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="imageCRSRef" type="gml:ImageCRSRefType" substitutionGroup="gml:coordinateReferenceSystemRef"/>
<!-- ===== -->
<complexType name="ImageCRSRefType">
  <annotation>
    <documentation>Association to an image coordinate reference system, either referencing or containing the definition of that reference system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateReferenceSystemRefType">
      <sequence>
        <element ref="gml:ImageCRS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="TemporalCRS" type="gml:TemporalCRSType" substitutionGroup="gml:_CoordinateReferenceSystem"/>
<!-- ===== -->
<complexType name="TemporalCRSType">
  <annotation>
    <documentation>A 1D coordinate reference system used for the recording of time. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateReferenceSystemType">
      <sequence>
        <element ref="gml:usesTemporalCS"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

        <element ref="gml:usesTemporalDatum"/>
    </sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="usesTemporalCS" type="gml:TemporalCSRefType">
    <annotation>
        <documentation>Association to the temporal coordinate system used by this CRS. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="usesTemporalDatum" type="gml:TemporalDatumRefType">
    <annotation>
        <documentation>Association to the temporal datum used by this CRS. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="temporalCRSRef" type="gml:TemporalCRSRefType"
substitutionGroup="gml:coordinateReferenceSystemRef"/>
<!-- ===== -->
<complexType name="TemporalCRSRefType">
    <annotation>
        <documentation>Association to a temporal coordinate reference system, either referencing or containing the definition of that reference system. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:CoordinateReferenceSystemRefType">
            <sequence>
                <element ref="gml:TemporalCRS" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
</schema>
```

## C.4 coordinateSystems.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" version="3.1.0" xml:lang="en">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-coordinateSystems"/>
        <documentation>
            <name>coordinateSystems.xsd</name>
            <version>3.1.0</version>
            <scope>How to encode coordinate system definitions. </scope>
            <description>Builds on referenceSystems.xsd to encode the data needed to define coordinate systems, including the specific subtypes of coordinate systems. Primary editor: Arliss Whiteside. Last updated 2003/10/16. </description>
            <copyright>Copyright (c) 2002-2003 OpenGIS, All Rights Reserved.</copyright>
            <conformance>This schema encodes the Coordinate System (CS_) package of the extended UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex C of Topic 2. </conformance>
            </documentation>
        </annotation>
        <!-- ===== -->
        includes and imports
        <!-- ===== -->
        <include schemaLocation="referenceSystems.xsd"/>
        <!-- ===== -->
        elements and types
        <!-- ===== -->
        <element name="CoordinateSystemAxis" type="gml:CoordinateSystemAxisType" substitutionGroup="gml:Definition"/>
        <!-- ===== -->
        <complexType name="CoordinateSystemAxisBaseType" abstract="true">
            <annotation>
```

```

<documentation>Basic encoding for coordinate system axis objects, simplifying and restricting the DefinitionType as
needed.</documentation>
</annotation>
<complexContent>
  <restriction base="gml:DefinitionType">
    <sequence>
      <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:remarks" minOccurs="0">
        <annotation>
          <documentation>Comments on or information about this coordinate system axis, including data
source information.</documentation>
        </annotation>
      </element>
      <element ref="gml:axisName"/>
    </sequence>
    <attribute ref="gml:id" use="required"/>
  </restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="axisName" type="gml:SimpleNameType" substitutionGroup="gml:name">
  <annotation>
    <documentation>The name by which this coordinate system axis is identified.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="CoordinateSystemAxisType">
  <annotation>
    <documentation>Definition of a coordinate system axis.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:CoordinateSystemAxisBaseType">
      <sequence>
        <element ref="gml:axisID" minOccurs="0" maxOccurs="unbounded">
          <annotation>
            <documentation>Set of alternative identifications of this coordinate system axis. The first axisID, if
any, is normally the primary identification code, and any others are aliases.</documentation>
          </annotation>
        </element>
        <element ref="gml:axisAbbrev"/>
        <element ref="gml:axisDirection"/>
      </sequence>
      <attribute ref="gml:uom" use="required"/>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="axisID" type="gml:IdentifierType">
  <annotation>
    <documentation>An identification of a coordinate system axis.</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="axisAbbrev" type="gml:CodeType">
  <annotation>
    <documentation>The abbreviation used for this coordinate system axis. This abbreviation can be used to identify the
ordinates in a coordinate tuple. Examples are X and Y. The codeSpace attribute can reference a source of more information on a
set of standardized abbreviations, or on this abbreviation.</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="axisDirection" type="gml:CodeType">
  <annotation>
    <documentation>Direction of this coordinate system axis (or in the case of Cartesian projected coordinates, the
direction of this coordinate system axis at the origin). Examples: north or south, east or west, up or down. Within any set of
coordinate system axes, only one of each pair of terms can be used. For earth-fixed CRSs, this direction is often approximate and
intended to provide a human interpretable meaning to the axis. When a geodetic datum is used, the precise directions of the axes
may therefore vary slightly from this approximate direction. Note that an EngineeringCRS can include specific descriptions of the
directions of its coordinate system axes. For example, the path of a linear CRS axis can be referenced in another document, such
as referencing a GML feature that references or includes a curve geometry. The codeSpace attribute can reference a source of
more information on a set of standardized directions, or on this direction.</documentation>
  </annotation>
</element>

```

```

        </annotation>
    </element>
    <!-- ===== -->
    <attribute name="uom" type="anyURI">
        <annotation>
            <documentation>Identifier of the unit of measure used for this coordinate system axis. The value of this coordinate in a coordinate tuple shall be recorded using this unit of measure, whenever those coordinates use a coordinate reference system that uses a coordinate system that uses this axis.</documentation>
        </annotation>
    </attribute>
    <!-- ===== -->
    <element name="coordinateSystemAxisRef" type="gml:CoordinateSystemAxisRefType" substitutionGroup="gml:dictionaryEntry"/>
    <!-- ===== -->
    <complexType name="CoordinateSystemAxisRefType">
        <annotation>
            <documentation>Association to a coordinate system axis, either referencing or containing the definition of that axis.</documentation>
        </annotation>
        <complexContent>
            <restriction base="gml:DictionaryEntryType">
                <sequence>
                    <element ref="gml:CoordinateSystemAxis" minOccurs="0"/>
                </sequence>
                <attributeGroup ref="gml:AssociationAttributeGroup"/>
            </restriction>
        </complexContent>
    </complexType>
    <!-- ===== -->
    <!-- ===== -->
    <element name="_CoordinateSystem" type="gml:AbstractCoordinateSystemType" abstract="true" substitutionGroup="gml:Definition"/>
    <!-- ===== -->
    <complexType name="AbstractCoordinateSystemBaseType" abstract="true">
        <annotation>
            <documentation>Basic encoding for coordinate system objects, simplifying and restricting the DefinitionType as needed.</documentation>
        </annotation>
        <complexContent>
            <restriction base="gml:DefinitionType">
                <sequence>
                    <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
                    <element ref="gml:remarks" minOccurs="0">
                        <annotation>
                            <documentation>Comments on or information about this coordinate system, including data source information.</documentation>
                        </annotation>
                    </element>
                    <element ref="gml:csName"/>
                </sequence>
                <attribute ref="gml:id" use="required"/>
            </restriction>
        </complexContent>
    </complexType>
    <!-- ===== -->
    <element name="csName" type="gml:SimpleNameType" substitutionGroup="gml:name">
        <annotation>
            <documentation>The name by which this coordinate system is identified.</documentation>
        </annotation>
    </element>
    <!-- ===== -->
    <complexType name="AbstractCoordinateSystemType" abstract="true">
        <annotation>
            <documentation>A coordinate system (CS) is the set of coordinate system axes that spans a given coordinate space. A CS is derived from a set of (mathematical) rules for specifying how coordinates in a given space are to be assigned to points. The coordinate values in a coordinate tuple shall be recorded in the order in which the coordinate system axes associations are recorded, whenever those coordinates use a coordinate reference system that uses this coordinate system. This abstract complexType shall not be used, extended, or restricted, in an Application Schema, to define a concrete subtype with a meaning equivalent to a concrete subtype specified in this document.</documentation>

```

```

</annotation>
<complexContent>
  <extension base="gml:AbstractCoordinateSystemBaseType">
    <sequence>
      <element ref="gml:csID" minOccurs="0" maxOccurs="unbounded">
        <annotation>
          <documentation>Set of alternative identifications of this coordinate system. The first csID, if any, is normally the primary identification code, and any others are aliases. </documentation>
        </annotation>
      </element>
      <element ref="gml:usesAxis" maxOccurs="unbounded">
        <annotation>
          <documentation>Ordered sequence of associations to the coordinate system axes included in this coordinate system. </documentation>
        </annotation>
      </element>
    </sequence>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="csID" type="gml:IdentifierType">
  <annotation>
    <documentation>An identification of a coordinate system. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="usesAxis" type="gml:CoordinateSystemAxisRefType" substitutionGroup="gml:dictionaryEntry">
  <annotation>
    <documentation>Association to a coordinate system axis. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="coordinateSystemRef" type="gml:CoordinateSystemRefType"/>
<!-- ===== -->
<complexType name="CoordinateSystemRefType">
  <annotation>
    <documentation>Association to a coordinate system, either referencing or containing the definition of that coordinate system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:_CoordinateSystem" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="EllipsoidalCS" type="gml:EllipsoidalCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- ===== -->
<complexType name="EllipsoidalCSType">
  <annotation>
    <documentation>A two- or three-dimensional coordinate system in which position is specified by geodetic latitude, geodetic longitude, and (in the three-dimensional case) ellipsoidal height. An EllipsoidalCS shall have two or three usesAxis associations. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="ellipsoidalCSRef" type="gml:EllipsoidalCSRefType" substitutionGroup="gml:coordinateSystemRef"/>
<!-- ===== -->
<complexType name="EllipsoidalCSRefType">
  <annotation>
    <documentation>Association to an ellipsoidal coordinate system, either referencing or containing the definition of that coordinate system. </documentation>
  </annotation>
  <complexContent>

```

```

<restriction base="gml:CoordinateSystemRefType">
  <sequence>
    <element ref="gml:EllipsoidalCS" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="CartesianCS" type="gml:CartesianCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- ===== -->
<complexType name="CartesianCSType">
  <annotation>
    <documentation>A 1-, 2-, or 3-dimensional coordinate system. Gives the position of points relative to orthogonal straight axes in the 2- and 3-dimensional cases. In the 1-dimensional case, it contains a single straight coordinate axis. In the multi-dimensional case, all axes shall have the same length unit of measure. A CartesianCS shall have one, two, or three usesAxis associations. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="cartesianCSRef" type="gml:CartesianCSRefType" substitutionGroup="gml:coordinateSystemRef"/>
<!-- ===== -->
<complexType name="CartesianCSRefType">
  <annotation>
    <documentation>Association to a Cartesian coordinate system, either referencing or containing the definition of that coordinate system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:CartesianCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="VerticalCS" type="gml:VerticalCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- ===== -->
<complexType name="VerticalCSType">
  <annotation>
    <documentation>A one-dimensional coordinate system used to record the heights (or depths) of points. Such a coordinate system is usually dependent on the Earth's gravity field, perhaps loosely as when atmospheric pressure is the basis for the vertical coordinate system axis. A VerticalCS shall have one usesAxis association. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="verticalCSRef" type="gml:VerticalCSRefType" substitutionGroup="gml:coordinateSystemRef"/>
<!-- ===== -->
<complexType name="VerticalCSRefType">
  <annotation>
    <documentation>Association to a vertical coordinate system, either referencing or containing the definition of that coordinate system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:VerticalCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

```

<!-- ===== -->
<element name="TemporalCS" type="gml:TemporalCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- ===== -->
<complexType name="TemporalCSType">
  <annotation>
    <documentation>A one-dimensional coordinate system containing a single time axis, used to describe the temporal position of a point in the specified time units from a specified time origin. A TemporalCS shall have one usesAxis association.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="temporalCSRef" type="gml:TemporalCSRefType" substitutionGroup="gml:coordinateSystemRef"/>
<!-- ===== -->
<complexType name="TemporalCSRefType">
  <annotation>
    <documentation>Association to a temporal coordinate system, either referencing or containing the definition of that coordinate system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:TemporalCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="LinearCS" type="gml:LinearCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- ===== -->
<complexType name="LinearCSType">
  <annotation>
    <documentation>A one-dimensional coordinate system that consists of the points that lie on the single axis described. The associated ordinate is the distance from the specified origin to the point along the axis. Example: usage of the line feature representing a road to describe points on or along that road. A LinearCS shall have one usesAxis association.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="linearCSRef" type="gml:LinearCSRefType" substitutionGroup="gml:coordinateSystemRef"/>
<!-- ===== -->
<complexType name="LinearCSRefType">
  <annotation>
    <documentation>Association to a linear coordinate system, either referencing or containing the definition of that coordinate system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:LinearCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="UserDefinedCS" type="gml:UserDefinedCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- ===== -->
<complexType name="UserDefinedCSType">
  <annotation>
    <documentation>A two- or three-dimensional coordinate system that consists of any combination of coordinate axes not covered by any other coordinate system type. An example is a multilinear coordinate system which contains one coordinate axis that may have any 1-D shape which has no intersections with itself. This non-straight axis is supplemented by one or two straight axes to complete a 2 or 3 dimensional coordinate system. The non-straight axis is typically incrementally straight or curved. A UserDefinedCS shall have two or three usesAxis associations. </documentation>
  </annotation>

```

```

</annotation>
<complexContent>
  <extension base="gml:AbstractCoordinateSystemType"/>
</complexContent>
</complexType>
<!-- ===== -->
<element name="userDefinedCSRef" type="gml:UserDefinedCSRefType" substitutionGroup="gml:coordinateSystemRef"/>
<!-- ===== -->
<complexType name="UserDefinedCSRefType">
  <annotation>
    <documentation>Association to a user-defined coordinate system, either referencing or containing the definition of that coordinate system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:UserDefinedCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="SphericalCS" type="gml:SphericalCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- ===== -->
<complexType name="SphericalCSType">
  <annotation>
    <documentation>A three-dimensional coordinate system with one distance measured from the origin and two angular coordinates. Not to be confused with an ellipsoidal coordinate system based on an ellipsoid "degenerated" into a sphere. A SphericalCS shall have three usesAxis associations. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="sphericalCSRef" type="gml:SphericalCSRefType" substitutionGroup="gml:coordinateSystemRef"/>
<!-- ===== -->
<complexType name="SphericalCSRefType">
  <annotation>
    <documentation>Association to a spherical coordinate system, either referencing or containing the definition of that coordinate system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:SphericalCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="PolarCS" type="gml:PolarCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- ===== -->
<complexType name="PolarCSType">
  <annotation>
    <documentation>A two-dimensional coordinate system in which position is specified by the distance from the origin and the angle between the line from the origin to a point and a reference direction. A PolarCS shall have two usesAxis associations. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="polarCSRef" type="gml:PolarCSRefType" substitutionGroup="gml:coordinateSystemRef"/>
<!-- ===== -->
<complexType name="PolarCSRefType">

```

```

<annotation>
  <documentation>Association to a polar coordinate system, either referencing or containing the definition of that
coordinate system. </documentation>
</annotation>
<complexContent>
  <restriction base="gml:CoordinateSystemRefType">
    <sequence>
      <element ref="gml:PolarCS" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </restriction>
</complexContent>
</complexType>
<!-- =====-->
<element name="CylindricalCS" type="gml:CylindricalCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- =====-->
<complexType name="CylindricalCSType">
  <annotation>
    <documentation>A three-dimensional coordinate system consisting of a polar coordinate system extended by a
straight coordinate axis perpendicular to the plane spanned by the polar coordinate system. A CylindricalCS shall have three
usesAxis associations. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
<!-- =====-->
<element name="cylindricalCSRef" type="gml:CylindricalCSRefType" substitutionGroup="gml:coordinateSystemRef"/>
<!-- =====-->
<complexType name="CylindricalCSRefType">
  <annotation>
    <documentation>Association to a cylindrical coordinate system, either referencing or containing the definition of that
coordinate system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:CylindricalCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- =====-->
<element name="ObliqueCartesianCS" type="gml:ObliqueCartesianCSType" substitutionGroup="gml:_CoordinateSystem"/>
<!-- =====-->
<complexType name="ObliqueCartesianCSType">
  <annotation>
    <documentation>A two- or three-dimensional coordinate system with straight axes that are not necessarily
orthogonal. An ObliqueCartesianCS shall have two or three usesAxis associations. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoordinateSystemType"/>
  </complexContent>
</complexType>
<!-- =====-->
<element name="obliqueCartesianCSRef" type="gml:ObliqueCartesianCSRefType"
substitutionGroup="gml:coordinateSystemRef"/>
<!-- =====-->
<complexType name="ObliqueCartesianCSRefType">
  <annotation>
    <documentation>Association to an oblique-Cartesian coordinate system, either referencing or containing the
definition of that coordinate system. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:CoordinateSystemRefType">
      <sequence>
        <element ref="gml:ObliqueCartesianCS" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>

```

```

        </complexContent>
    </complexType>
    <!-- ===== -->
</schema>
```

## C.5 coverage.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns:gml="http://www.opengis.net/gml"
       xmlns="http://www.w3.org/2001/XMLSchema" xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDefault="qualified"
       version="3.1.0">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-xsd:coverage:v3.1.0">coverage.xsd</appinfo>
        <documentation xml:lang="en"> GML Coverage schema.
        Copyright (c) 2004 OGC, All Rights Reserved.
    </annotation>
    <documentation>
        <!-- ===== -->
        includes and imports
        <!-- ===== -->
        <include schemaLocation="feature.xsd"/>
        <include schemaLocation="valueObjects.xsd"/>
        <include schemaLocation="grids.xsd"/>
        <include schemaLocation="geometryAggregates.xsd"/>
        <!-- ===== -->
        global types and elements
        <!-- ===== -->
        <!-- ===== Abstract coverage definition ===== -->
        <!-- ===== -->
        <!-- ===== -->
        <element name="_Coverage" type="gml:AbstractCoverageType" abstract="true" substitutionGroup="gml:_Feature"/>
        <!-- ===== -->
        <complexType name="AbstractCoverageType" abstract="true">
            <annotation>
                <documentation>Abstract element which acts as the head of a substitution group for coverages. Note that a coverage
is a GML feature. </documentation>
            </annotation>
            <complexContent>
                <extension base="gml:AbstractFeatureType">
                    <sequence>
                        <element ref="gml:domainSet"/>
                        <element ref="gml:rangeSet"/>
                    </sequence>
                    <attribute name="dimension" type="positiveInteger" use="optional"/>
                </extension>
            </complexContent>
        </complexType>
        <!-- ===== -->
        <element name="_ContinuousCoverage" type="gml:AbstractContinuousCoverageType" abstract="true"
substitutionGroup="gml:_Feature"/>
        <!-- ===== -->
        <complexType name="AbstractContinuousCoverageType" abstract="true">
            <annotation>
                <documentation>A continuous coverage as defined in ISO 19123 is a coverage that can return different values for the
same feature attribute at different direct positions within a single spatiotemporal object in its spatiotemporal
domain</documentation>
            </annotation>
            <complexContent>
                <extension base="gml:AbstractCoverageType">
                    <sequence>
                        <element ref="gml:coverageFunction" minOccurs="0"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
        <!-- ===== -->
        <element name="_DiscreteCoverage" type="gml:AbstractDiscreteCoverageType" abstract="true"
substitutionGroup="gml:_Coverage"/>
    </annotation>

```

```

<!-- ===== -->
<complexType name="AbstractDiscreteCoverageType" abstract="true">
  <annotation>
    <documentation>A discrete coverage consists of a domain set, range set and optionally a coverage function. The domain set consists of either geometry or temporal objects, finite in number. The range set is comprised of a finite number of attribute values each of which is associated to every direct position within any single spatiotemporal object in the domain. In other words, the range values are constant on each spatiotemporal object in the domain. This coverage function maps each element from the coverage domain to an element in its range. This definition conforms to ISO 19123.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCoverageType">
      <sequence>
        <element ref="gml:coverageFunction" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="domainSet" type="gml:DomainSetType"/>
<!-- ===== -->
<complexType name="DomainSetType">
  <annotation>
    <documentation>The spatiotemporal domain of a coverage.

```

Typically

- \* a geometry collection,
- \* an implicit geometry (e.g. a grid),
- \* an explicit or implicit collection of time instances or periods, or

N.B. Temporal geometric complexes and temporal grids are not yet implemented in GML.</documentation>

```

  </annotation>
  <choice minOccurs="0">
    <element ref="gml:_Geometry"/>
    <element ref="gml:_TimeObject"/>
  </choice>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<element name="rangeSet" type="gml:RangeSetType"/>
<!-- ===== -->
<complexType name="RangeSetType">
  <choice>
    <element ref="gml:ValueArray" maxOccurs="unbounded">
      <annotation>
        <documentation>each member _Value holds a tuple or "row" from the equivalent table</documentation>
      </annotation>
    </element>
    <element ref="gml:_ScalarValueList" maxOccurs="unbounded">
      <annotation>
        <documentation>each list holds the complete set of one scalar component from the values - i.e. a "column" from the equivalent table</documentation>
      </annotation>
    </element>
    <element ref="gml:DataBlock">
      <annotation>
        <documentation>Its tuple list holds the values as space-separated tuples each of which contains comma-separated components, and the tuple structure is specified using the rangeParameters property.</documentation>
      </annotation>
    </element>
    <element ref="gml:File">
      <annotation>
        <documentation>a reference to an external source for the data, together with a description of how that external source is structured</documentation>
      </annotation>
    </element>
  </choice>
</complexType>
<!-- ===== -->
<element name="coverageFunction" type="gml:Cov
```

```

<complexType name="CoverageFunctionType">
  <annotation>
    <documentation>
      The function or rule which defines the map from members of the domainSet to the range.
      More functions will be added to this list
    </documentation>
  </annotation>
  <choice>
    <element ref="gml:MappingRule"/>
    <element ref="gml:GridFunction"/>
  </choice>
</complexType>
<!-- ===== Components for encoding the rangeSet ===== -->
<!-- ===== Components for coverageFunctions ===== -->
<element name="DataBlock" type="gml:DataBlockType"/>
<!-- ===== -->
<complexType name="DataBlockType">
  <sequence>
    <element ref="gml:rangeParameters"/>
    <choice>
      <element ref="gml:tupleList"/>
      <element ref="gml:doubleOrNullTupleList"/>
    </choice>
  </sequence>
</complexType>
<!-- ===== -->
<element name="tupleList" type="gml:CoordinatesType"/>
<!-- ===== -->
<element name="doubleOrNullTupleList" type="gml:doubleOrNullList"/>
<!-- ===== -->
<element name="File" type="gml:FileType"/>
<!-- ===== -->
<complexType name="FileType">
  <sequence>
    <element ref="gml:rangeParameters"/>
    <element name="fileName" type="anyURI"/>
    <element name="fileStructure" type="gml:FileValueModelType"/>
    <element name="mimeType" type="anyURI" minOccurs="0"/>
    <element name="compression" type="anyURI" minOccurs="0"/>
  </sequence>
</complexType>
<!-- ===== -->
<simpleType name="FileValueModelType">
  <annotation>
    <documentation>List of codes that identifies the file structure model for records stored in files.</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="Record Interleaved"/>
  </restriction>
</simpleType>
<!-- ===== -->
<element name="rangeParameters" type="gml:RangeParametersType"/>
<!-- ===== -->
<complexType name="RangeParametersType">
  <annotation>
    <documentation>
      Metadata about the rangeSet. Definition of record structure.
      This is required if the rangeSet is encoded in a DataBlock.
      We use a gml:_Value with empty values as a map of the composite value structure.</documentation>
    </annotation>
    <sequence minOccurs="0">
      <element ref="gml:_Value"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </complexType>
<!-- ===== -->
<!-- ===== Components for coverageFunctions ===== -->
<!-- ===== -->

```

```

<element name="MappingRule" type="gml:StringOrRefType">
  <annotation>
    <documentation>Description of a rule for associating members from the domainSet with members of the rangeSet.</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="GridFunction" type="gml:GridFunctionType"/>
<!-- ===== -->
<complexType name="GridFunctionType">
  <annotation>
    <documentation>Defines how values in the domain are mapped to the range set. The start point and the sequencing rule are specified here.</documentation>
  </annotation>
  <sequence>
    <element name="sequenceRule" type="gml:SequenceRuleType" minOccurs="0">
      <annotation>
        <documentation>If absent, the implied value is "Linear".</documentation>
      </annotation>
    </element>
    <element name="startPoint" type="gml:integerList" minOccurs="0">
      <annotation>
        <documentation>Index position of the first grid post, which must lie somewhere in the GridEnvelope. If absent, the startPoint is equal to the value of gridEnvelope::low from the grid definition.</documentation>
      </annotation>
    </element>
  </sequence>
</complexType>
<!-- ===== -->
<element name="IndexMap" type="gml:IndexMapType" substitutionGroup="gml:GridFunction"/>
<!-- ===== -->
<complexType name="IndexMapType">
  <annotation>
    <documentation>Exends GridFunctionType with a lookUpTable. This contains a list of indexes of members within the rangeSet corresponding with the members of the domainSet. The domainSet is traversed in list order if it is enumerated explicitly, or in the order specified by a SequenceRule if the domain is an implicit set. The length of the lookUpTable corresponds with the length of the subset of the domainSet for which the coverage is defined.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:GridFunctionType">
      <sequence>
        <element name="lookUpTable" type="gml:integerList"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<complexType name="SequenceRuleType">
  <simpleContent>
    <extension base="gml:SequenceRuleNames">
      <attribute name="order" type="gml:IncrementOrder" use="optional"/>
    </extension>
  </simpleContent>
</complexType>
<!-- ===== -->
<simpleType name="SequenceRuleNames">
  <annotation>
    <documentation>List of codes (adopted from ISO 19123 Annex C) that identifies the rule for traversing a grid to correspond with the sequence of members of the rangeSet.</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="Linear"/>
    <enumeration value="Boustrophedonic"/>
    <enumeration value="Cantor-diagonal"/>
    <enumeration value="Spiral"/>
    <enumeration value="Morton"/>
    <enumeration value="Hilbert"/>
  </restriction>
</simpleType>
<!-- ===== -->
<simpleType name="IncrementOrder">

```

```

<annotation>
  <documentation>The enumeration value here indicates the incrementation order to be used on the first 2 axes, i.e. "+x-y" means that the points on the first axis are to be traversed from lowest to highest and the points on the second axis are to be traversed from highest to lowest. The points on all other axes (if any) beyond the first 2 are assumed to increment from lowest to highest.</documentation>
</annotation>
<restriction base="string">
  <enumeration value="+x+y"/>
  <enumeration value="+y+x"/>
  <enumeration value="+x-y"/>
  <enumeration value="-x-y"/>
</restriction>
</simpleType>
<!-- ===== -->
<!-- == Specialised Coverage types - typed by the structure of the domain set == -->
<!-- ===== -->
<element name="MultiPointCoverage" type="gml:MultiPointCoverageType" substitutionGroup="gml:_DiscreteCoverage"/>
<!-- ===== -->
<complexType name="MultiPointCoverageType">
  <annotation>
    <documentation>A discrete coverage type whose domain is defined by a collection of point</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:AbstractDiscreteCoverageType">
      <sequence>
        <group ref="gml:StandardObjectProperties"/>
        <element ref="gml:boundedBy" minOccurs="0"/>
        <element ref="gml:multiPointDomain"/>
        <element ref="gml:rangeSet"/>
        <element ref="gml:coverageFunction" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="multiPointDomain" type="gml:MultiPointDomainType" substitutionGroup="gml:domainSet"/>
<!-- ===== -->
<complexType name="MultiPointDomainType">
  <complexContent>
    <restriction base="gml:DomainSetType">
      <sequence minOccurs="0">
        <element ref="gml:MultiPoint"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="MultiCurveCoverage" type="gml:MultiCurveCoverageType" substitutionGroup="gml:_DiscreteCoverage"/>
<!-- ===== -->
<complexType name="MultiCurveCoverageType">
  <annotation>
    <documentation>A discrete coverage type whose domain is defined by a collection of curves.</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:AbstractDiscreteCoverageType">
      <sequence>
        <group ref="gml:StandardObjectProperties"/>
        <element ref="gml:boundedBy" minOccurs="0"/>
        <element ref="gml:multiCurveDomain"/>
        <element ref="gml:rangeSet"/>
        <element ref="gml:coverageFunction" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="multiCurveDomain" type="gml:MultiCurveDomainType" substitutionGroup="gml:domainSet"/>

```

```

<!-- ===== -->
<complexType name="MultiCurveDomainType">
  <complexContent>
    <restriction base="gml:DomainSetType">
      <sequence minOccurs="0">
        <element ref="gml:MultiCurve"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="MultiSurfaceCoverage" type="gml:MultiSurfaceCoverageType"
substitutionGroup="gml:_DiscreteCoverage"/>
<!-- ===== -->
<complexType name="MultiSurfaceCoverageType">
  <annotation>
    <documentation>A discrete coverage type whose domain is defined by a collection of surface patches (includes polygons, triangles, rectangles, etc).</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:AbstractDiscreteCoverageType">
      <sequence>
        <group ref="gml:StandardObjectProperties"/>
        <element ref="gml:boundedBy" minOccurs="0"/>
        <element ref="gml:multiSurfaceDomain"/>
        <element ref="gml:rangeSet"/>
        <element ref="gml:coverageFunction" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="multiSurfaceDomain" type="gml:MultiSurfaceDomainType" substitutionGroup="gml:domainSet"/>
<!-- ===== -->
<complexType name="MultiSurfaceDomainType">
  <complexContent>
    <restriction base="gml:DomainSetType">
      <sequence minOccurs="0">
        <element ref="gml:MultiSurface"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="MultiSolidCoverage" type="gml:MultiSolidCoverageType" substitutionGroup="gml:_DiscreteCoverage"/>
<!-- ===== -->
<complexType name="MultiSolidCoverageType">
  <annotation>
    <documentation>A discrete coverage type whose domain is defined by a collection of Solids.</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:AbstractDiscreteCoverageType">
      <sequence>
        <group ref="gml:StandardObjectProperties"/>
        <element ref="gml:boundedBy" minOccurs="0"/>
        <element ref="gml:multiSolidDomain"/>
        <element ref="gml:rangeSet"/>
        <element ref="gml:coverageFunction" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="multiSolidDomain" type="gml:MultiSolidDomainType" substitutionGroup="gml:domainSet"/>
<!-- ===== -->
<complexType name="MultiSolidDomainType">
  <complexContent>
    <restriction base="gml:DomainSetType">
      <sequence minOccurs="0">

```

```

        <element ref="gml:MultiSolid"/>
    </sequence>
</restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="GridCoverage" type="gml:GridCoverageType" substitutionGroup="gml:_DiscreteCoverage"/>
<!-- ===== -->
<complexType name="GridCoverageType">
    <complexContent>
        <restriction base="gml:AbstractDiscreteCoverageType">
            <sequence>
                <group ref="gml:StandardObjectProperties"/>
                <element ref="gml:boundedBy" minOccurs="0"/>
                <element ref="gml:gridDomain"/>
                <element ref="gml:rangeSet"/>
                <element ref="gml:coverageFunction" minOccurs="0"/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="gridDomain" type="gml:GridDomainType" substitutionGroup="gml:domainSet"/>
<!-- ===== -->
<complexType name="GridDomainType">
    <complexContent>
        <restriction base="gml:DomainSetType">
            <choice minOccurs="0">
                <element ref="gml:Grid"/>
            </choice>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="RectifiedGridCoverage" type="gml:RectifiedGridCoverageType"
substitutionGroup="gml:_DiscreteCoverage"/>
<!-- ===== -->
<complexType name="RectifiedGridCoverageType">
    <complexContent>
        <restriction base="gml:AbstractDiscreteCoverageType">
            <sequence>
                <group ref="gml:StandardObjectProperties"/>
                <element ref="gml:boundedBy" minOccurs="0"/>
                <element ref="gml:rectifiedGridDomain"/>
                <element ref="gml:rangeSet"/>
                <element ref="gml:coverageFunction" minOccurs="0"/>
            </sequence>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="rectifiedGridDomain" type="gml:RectifiedGridDomainType" substitutionGroup="gml:domainSet"/>
<!-- ===== -->
<complexType name="RectifiedGridDomainType">
    <complexContent>
        <restriction base="gml:DomainSetType">
            <choice minOccurs="0">
                <element ref="gml:RectifiedGrid"/>
            </choice>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
</schema>
```

## C.6 dataQuality.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.4 U (http://www.xmlspy.com) by Arliss J Whiteside (BAE Systems) -->
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" version="3.1.0" xml:lang="en">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-dataQuality"/>
    <documentation>
      <name>dataQuality.xsd</name>
      <version>3.1.0</version>
      <scope>How to encode positional data quality information. </scope>
      <description>Builds on units.xsd to encode the data needed to describe the positional accuracy of coordinate operations. Primary editor: Arliss Whiteside. Last updated 2003/10/16. </description>
      <copyright>Copyright (c) 2002-2003 OpenGIS, All Rights Reserved.</copyright>
      <conformance>This schema encodes the Data Quality (DQ) package of the extended UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex C of Topic 2. </conformance>
    </documentation>
  </annotation>
  <!-- =====
  includes and imports
  ===== -->
  <include schemaLocation="units.xsd"/>
  <!-- =====
  elements and types
  ===== -->
  <element name="_positionalAccuracy" type="gml:AbstractPositionalAccuracyType" abstract="true"/>
  <!-- =====
  <complexType name="AbstractPositionalAccuracyType" abstract="true">
    <annotation>
      <documentation>Position error estimate (or accuracy) data. </documentation>
    </annotation>
    <sequence>
      <element ref="gml:measureDescription" minOccurs="0"/>
    </sequence>
  </complexType>
  <!-- =====
  <element name="measureDescription" type="gml:CodeType">
    <annotation>
      <documentation>A description of the position accuracy parameter(s) provided. </documentation>
    </annotation>
  </element>
  <!-- =====
  <element name="absoluteExternalPositionalAccuracy" type="gml:AbsoluteExternalPositionalAccuracyType"
substitutionGroup="gml:_positionalAccuracy"/>
  <!-- =====
  <complexType name="AbsoluteExternalPositionalAccuracyType">
    <annotation>
      <documentation>Closeness of reported coordinate values to values accepted as or being true. </documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractPositionalAccuracyType">
        <sequence>
          <element ref="gml:result"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <!-- =====
  <element name="relativeInternalPositionalAccuracy" type="gml:RelativeInternalPositionalAccuracyType"
substitutionGroup="gml:_positionalAccuracy"/>
  <!-- =====
  <complexType name="RelativeInternalPositionalAccuracyType">
    <annotation>
      <documentation>Closeness of the relative positions of two or more positions to their respective relative positions accepted as or being true. </documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractPositionalAccuracyType">

```

```

        <sequence>
            <element ref="gml:result"/>
        </sequence>
    </extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="result" type="gml:MeasureType">
    <annotation>
        <documentation>A quantitative result defined by the evaluation procedure used, and identified by the measureDescription. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="covarianceMatrix" type="gml:CovarianceMatrixType" substitutionGroup="gml:_positionalAccuracy"/>
<!-- ===== -->
<complexType name="CovarianceMatrixType">
    <annotation>
        <documentation>Error estimate covariance matrix. </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractPositionalAccuracyType">
            <sequence>
                <element ref="gml:unitOfMeasure" maxOccurs="unbounded">
                    <annotation>
                        <documentation>Ordered sequence of units of measure, corresponding to the row and column index numbers of the covariance matrix, starting with row and column 1 and ending with row/column N. Each unit of measure is for the ordinate reflected in the relevant row and column of the covariance matrix. </documentation>
                    </annotation>
                </element>
                <element ref="gml:includesElement" maxOccurs="unbounded">
                    <annotation>
                        <documentation>Unordered set of elements in this covariance matrix. Because the covariance matrix is symmetrical, only the elements in the upper or lower diagonal part (including the main diagonal) of the matrix need to be specified. Any zero valued covariance elements can be omitted. </documentation>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="includesElement" type="gml:CovarianceElementType"/>
<!-- ===== -->
<complexType name="CovarianceElementType">
    <annotation>
        <documentation>An element of a covariance matrix. </documentation>
    </annotation>
    <sequence>
        <element ref="gml:rowIndex"/>
        <element ref="gml:columnIndex"/>
        <element ref="gml:covariance"/>
    </sequence>
</complexType>
<!-- ===== -->
<element name="rowIndex" type="positiveInteger">
    <annotation>
        <documentation>Row number of this covariance element value. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="columnIndex" type="positiveInteger">
    <annotation>
        <documentation>Column number of this covariance element value. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="covariance" type="double">
    <annotation>

```

```

        <documentation>Value of covariance matrix element. </documentation>
    </annotation>
</element>
<!-- ===== -->
</schema>
```

## C.7 datums.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
       xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" version="3.1.0" xml:lang="en">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-datums"/>
        <documentation>
            <name>datums.xsd</name>
            <version>3.1.0</version>
            <scope>How to encode datum definitions. </scope>
            <description>Builds on referenceSystems.xsd to encode the data needed to define datums, including the specific subtypes of datums. Primary editor: Arliss Whiteside. Last updated 2003/10/16. </description>
            <copyright>Copyright (c) 2002-2003 Open GIS Consortium, All Rights Reserved.</copyright>
            <conformance>This schema encodes the Datum (CD_) package of the extended UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex C of Topic 2. </conformance>
        </documentation>
    </annotation>
    <!-- ===== -->
    includes and imports
    =====>
    <include schemaLocation="referenceSystems.xsd"/>
    <!-- ===== -->
    elements and types
    =====>
    <element name="_Datum" type="gml:AbstractDatumType" abstract="true" substitutionGroup="gml:Definition"/>
    <!-- ===== -->
    <complexType name="AbstractDatumBaseType" abstract="true">
        <annotation>
            <documentation>Basic encoding for datum objects, simplifying and restricting the DefinitionType as needed. </documentation>
        </annotation>
        <complexContent>
            <restriction base="gml:DefinitionType">
                <sequence>
                    <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
                    <element ref="gml:remarks" minOccurs="0">
                        <annotation>
                            <documentation>Comments on this reference system, including source information. </documentation>
                        </annotation>
                    </element>
                    <element ref="gml:datumName"/>
                </sequence>
                <attribute ref="gml:id" use="required"/>
            </restriction>
        </complexContent>
    </complexType>
    <!-- ===== -->
    <element name="datumName" type="gml:SimpleNameType" substitutionGroup="gml:name">
        <annotation>
            <documentation>The name by which this datum is identified. </documentation>
        </annotation>
    </element>
    <!-- ===== -->
    <complexType name="AbstractDatumType" abstract="true">
        <annotation>
            <documentation>A datum specifies the relationship of a coordinate system to the earth, thus creating a coordinate reference system. A datum uses a parameter or set of parameters that determine the location of the origin of the coordinate reference system. Each datum subtype can be associated with only specific types of coordinate systems. This abstract complexType shall not be used, extended, or restricted, in an Application Schema, to define a concrete subtype with a meaning equivalent to a concrete subtype specified in this document. </documentation>
        </annotation>

```

</annotation>

```

<complexContent>
  <extension base="gml:AbstractDatumBaseType">
    <sequence>
      <element ref="gml:datumID" minOccurs="0" maxOccurs="unbounded">
        <annotation>
          <documentation>Set of alternative identifications of this datum. The first datumID, if any, is normally the primary identification code, and any others are aliases. </documentation>
        </annotation>
      </element>
      <element ref="gml:anchorPoint" minOccurs="0"/>
      <element ref="gml:realizationEpoch" minOccurs="0"/>
      <element ref="gml:validArea" minOccurs="0"/>
      <element ref="gml:scope" minOccurs="0"/>
    </sequence>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="datumID" type="gml:IdentifierType">
  <annotation>
    <documentation>An identification of a datum. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="anchorPoint" type="gml:CodeType">
  <annotation>
    <documentation>Description, possibly including coordinates, of the point or points used to anchor the datum to the Earth. Also known as the "origin", especially for engineering and image datums. The codeSpace attribute can be used to reference a source of more detailed on this point or surface, or on a set of such descriptions.
      - For a geodetic datum, this point is also known as the fundamental point, which is traditionally the point where the relationship between geoid and ellipsoid is defined. In some cases, the "fundamental point" may consist of a number of points. In those cases, the parameters defining the geoid/ellipsoid relationship have been averaged for these points, and the averages adopted as the datum definition.
      - For an engineering datum, the anchor point may be a physical point, or it may be a point with defined coordinates in another CRS. When appropriate, the coordinates of this anchor point can be referenced in another document, such as referencing a GML feature that references or includes a point position.
      - For an image datum, the anchor point is usually either the centre of the image or the corner of the image.
      - For a temporal datum, this attribute is not defined. Instead of the anchor point, a temporal datum carries a separate time origin of type DateTime. </documentation>
    </annotation>
  </element>
<!-- ===== -->
<element name="realizationEpoch" type="date">
  <annotation>
    <documentation>The time after which this datum definition is valid. This time may be precise (e.g. 1997.0 for IRTF97) or merely a year (e.g. 1983 for NAD83). In the latter case, the epoch usually refers to the year in which a major recalculation of the geodetic control network, underlying the datum, was executed or initiated. An old datum can remain valid after a new datum is defined. Alternatively, a datum may be superseded by a later datum, in which case the realization epoch for the new datum defines the upper limit for the validity of the superseded datum. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="datumRef" type="gml:DatumRefType" substitutionGroup="gml:dictionaryEntry"/>
<!-- ===== -->
<complexType name="DatumRefType">
  <annotation>
    <documentation>Association to a datum, either referencing or containing the definition of that datum. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:_Datum" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->

```

```

<element name="EngineeringDatum" type="gml:EngineeringDatumType" substitutionGroup="gml:_Datum"/>
<!-- ===== -->
<complexType name="EngineeringDatumType">
  <annotation>
    <documentation>An engineering datum defines the origin of an engineering coordinate reference system, and is used in a region around that origin. This origin can be fixed with respect to the earth (such as a defined point at a construction site), or be a defined point on a moving vehicle (such as on a ship or satellite). </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractDatumType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="engineeringDatumRef" type="gml:EngineeringDatumRefType" substitutionGroup="gml:datumRef"/>
<!-- ===== -->
<complexType name="EngineeringDatumRefType">
  <annotation>
    <documentation>Association to an engineering datum, either referencing or containing the definition of that datum. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DatumRefType">
      <sequence>
        <element ref="gml:EngineeringDatum" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="ImageDatum" type="gml:ImageDatumType" substitutionGroup="gml:_Datum"/>
<!-- ===== -->
<complexType name="ImageDatumType">
  <annotation>
    <documentation>An image datum defines the origin of an image coordinate reference system, and is used in a local context only. For more information, see OGC Abstract Specification Topic 2. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractDatumType">
      <sequence>
        <element ref="gml:pixelInCell"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="pixelInCell" type="gml:PixelInCellType"/>
<!-- ===== -->
<complexType name="PixelInCellType">
  <annotation>
    <documentation>Specification of the way an image grid is associated with the image data attributes. </documentation>
  </annotation>
  <simpleContent>
    <restriction base="gml:CodeType">
      <attribute name="codeSpace" type="anyURI" use="required">
        <annotation>
          <documentation>Reference to a source of information specifying the values and meanings of all the allowed string values for this PixelInCellType. </documentation>
        </annotation>
      </attribute>
    </restriction>
  </simpleContent>
</complexType>
<!-- ===== -->
<element name="imageDatumRef" type="gml:ImageDatumRefType" substitutionGroup="gml:datumRef"/>
<!-- ===== -->
<complexType name="ImageDatumRefType">
  <annotation>
    <documentation>Association to an image datum, either referencing or containing the definition of that datum. </documentation>
  </annotation>
</complexType>

```

```

</annotation>
<complexContent>
  <restriction base="gml:DatumRefType">
    <sequence>
      <element ref="gml:ImageDatum" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="VerticalDatum" type="gml:VerticalDatumType" substitutionGroup="gml:_Datum"/>
<!-- ===== -->
<complexType name="VerticalDatumType">
  <annotation>
    <documentation>A textual description and/or a set of parameters identifying a particular reference level surface used as a zero-height surface, including its position with respect to the Earth for any of the height types recognized by this standard. There are several types of Vertical Datums, and each may place constraints on the Coordinate Axis with which it is combined to create a Vertical CRS. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractDatumType">
      <sequence>
        <element ref="gml:verticalDatumType" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="verticalDatumType" type="gml:VerticalDatumTypeType"/>
<!-- ===== -->
<complexType name="VerticalDatumTypeType">
  <annotation>
    <documentation>Type of a vertical datum. </documentation>
  </annotation>
  <simpleContent>
    <restriction base="gml:CodeType">
      <attribute name="codeSpace" type="anyURI" use="required">
        <annotation>
          <documentation>Reference to a source of information specifying the values and meanings of all the allowed string values for this VerticalDatumTypeType. </documentation>
        </annotation>
      </attribute>
    </restriction>
  </simpleContent>
</complexType>
<!-- ===== -->
<element name="verticalDatumRef" type="gml:VerticalDatumRefType" substitutionGroup="gml:datumRef"/>
<!-- ===== -->
<complexType name="VerticalDatumRefType">
  <annotation>
    <documentation>Association to a vertical datum, either referencing or containing the definition of that datum. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DatumRefType">
      <sequence>
        <element ref="gml:VerticalDatum" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="TemporalDatum" type="gml:TemporalDatumType" substitutionGroup="gml:_Datum"/>
<!-- ===== -->
<complexType name="TemporalDatumBaseType" abstract="true">
  <annotation>

```

<documentation>Partially defines the origin of a temporal coordinate reference system. This type restricts the AbstractDatumType to remove the "anchorPoint" and "realizationEpoch" elements. </documentation>

```

</annotation>
<complexContent>
  <restriction base="gml:AbstractDatumType">
    <sequence>
      <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:datumName"/>
      <element ref="gml:datumID" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:validArea" minOccurs="0"/>
      <element ref="gml:scope" minOccurs="0"/>
    </sequence>
    <attribute ref="gml:id" use="required"/>
  </restriction>
</complexContent>
</complexType>
<!-- ===== -->
<complexType name="TemporalDatumType">
  <annotation>
    <documentation>Defines the origin of a temporal coordinate reference system. This type extends the TemporalDatumRestrictionType to add the "origin" element with the dateTime type. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:TemporalDatumBaseType">
      <sequence>
        <element ref="gml:origin"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="origin" type="dateTime">
  <annotation>
    <documentation>The date and time origin of this temporal datum. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="temporalDatumRef" type="gml:TemporalDatumRefType" substitutionGroup="gml:datumRef"/>
<!-- ===== -->
<complexType name="TemporalDatumRefType">
  <annotation>
    <documentation>Association to a temporal datum, either referencing or containing the definition of that datum. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DatumRefType">
      <sequence>
        <element ref="gml:TemporalDatum" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="GeodeticDatum" type="gml:GeodeticDatumType" substitutionGroup="gml:_Datum"/>
<!-- ===== -->
<complexType name="GeodeticDatumType">
  <annotation>
    <documentation>A geodetic datum defines the precise location and orientation in 3-dimensional space of a defined ellipsoid (or sphere) that approximates the shape of the earth, or of a Cartesian coordinate system centered in this ellipsoid (or sphere). </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractDatumType">
      <sequence>
        <element ref="gml:usesPrimeMeridian"/>
        <element ref="gml:usesEllipsoid"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

<!-- ===== -->
<element name="usesPrimeMeridian" type="gml:PrimeMeridianRefType">
  <annotation>
    <documentation>Association to the prime meridian used by this geodetic datum. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="usesEllipsoid" type="gml:EllipsoidRefType">
  <annotation>
    <documentation>Association to the ellipsoid used by this geodetic datum. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="geodeticDatumRef" type="gml:GeodeticDatumRefType" substitutionGroup="gml:datumRef"/>
<!-- ===== -->
<complexType name="GeodeticDatumRefType">
  <annotation>
    <documentation>Association to a geodetic datum, either referencing or containing the definition of that datum. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DatumRefType">
      <sequence>
        <element ref="gml:GeodeticDatum" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="PrimeMeridian" type="gml:PrimeMeridianType" substitutionGroup="gml:Definition"/>
<!-- ===== -->
<complexType name="PrimeMeridian BaseType" abstract="true">
  <annotation>
    <documentation>Basic encoding for prime meridian objects, simplifying and restricting the DefinitionType as needed. </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DefinitionType">
      <sequence>
        <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:remarks" minOccurs="0">
          <annotation>
            <documentation>Comments on or information about this prime meridian, including source information. </documentation>
          </annotation>
        </element>
        <element ref="gml:meridianName"/>
      </sequence>
      <attribute ref="gml:id" use="required"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="meridianName" type="gml:SimpleNameType" substitutionGroup="gml:name">
  <annotation>
    <documentation>The name by which this prime meridian is identified. The meridianName most common value is "Greenwich", and that value shall be used when the greenwichLongitude value is zero. </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="PrimeMeridianType">
  <annotation>
    <documentation>A prime meridian defines the origin from which longitude values are determined.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:PrimeMeridian BaseType">
      <sequence>

```

```

<element ref="gml:meridianID" minOccurs="0" maxOccurs="unbounded">
    <annotation>
        <documentation>Set of alternative identifications of this prime meridian. The first meridianID, if any, is normally the primary identification code, and any others are aliases. </documentation>
    </annotation>
</element>
<element ref="gml:greenwichLongitude"/>
</sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="meridianID" type="gml:IdentifierType">
    <annotation>
        <documentation>An identification of a prime meridian. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="greenwichLongitude" type="gml:AngleChoiceType">
    <annotation>
        <documentation>Longitude of the prime meridian measured from the Greenwich meridian, positive eastward. The greenwichLongitude most common value is zero, and that value shall be used when the meridianName value is "Greenwich". </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="primeMeridianRef" type="gml:PrimeMeridianRefType" substitutionGroup="gml:dictionaryEntry"/>
<!-- ===== -->
<complexType name="PrimeMeridianRefType">
    <annotation>
        <documentation>Association to a prime meridian, either referencing or containing the definition of that meridian. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:DictionaryEntryType">
            <sequence>
                <element ref="gml:PrimeMeridian" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="Ellipsoid" type="gml:EllipsoidType" substitutionGroup="gml:Definition"/>
<!-- ===== -->
<complexType name="Ellipsoid BaseType" abstract="true">
    <annotation>
        <documentation>Basic encoding for ellipsoid objects, simplifying and restricting the DefinitionType as needed. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:DefinitionType">
            <sequence>
                <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:remarks" minOccurs="0">
                    <annotation>
                        <documentation>Comments on or information about this ellipsoid, including source information. </documentation>
                    </annotation>
                </element>
                <element ref="gml:ellipsoidName"/>
            </sequence>
            <attribute ref="gml:id" use="required"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="ellipsoidName" type="gml:SimpleNameType" substitutionGroup="gml:name">
    <annotation>
        <documentation>The name by which this ellipsoid is identified. </documentation>
    </annotation>
</element>

```

```

</element>
<!-- ===== -->
<complexType name="EllipsoidType">
  <annotation>
    <documentation>An ellipsoid is a geometric figure that can be used to describe the approximate shape of the earth. In mathematical terms, it is a surface formed by the rotation of an ellipse about its minor axis.</documentation>
    </annotation>
  <complexContent>
    <extension base="gml:Ellipsoid BaseType">
      <sequence>
        <element ref="gml:ellipsoidID" minOccurs="0" maxOccurs="unbounded">
          <annotation>
            <documentation>Set of alternative identifications of this ellipsoid. The first ellipsoidID, if any, is normally the primary identification code, and any others are aliases.</documentation>
            </annotation>
          </element>
          <element ref="gml:semiMajorAxis"/>
          <element ref="gml:secondDefiningParameter"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
<!-- ===== -->
<element name="ellipsoidID" type="gml:IdentifierType">
  <annotation>
    <documentation>An identification of an ellipsoid.</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="semiMajorAxis" type="gml:LengthType">
  <annotation>
    <documentation>Length of the semi-major axis of the ellipsoid.</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="ellipsoidRef" type="gml:EllipsoidRefType" substitutionGroup="gml:dictionaryEntry"/>
<!-- ===== -->
<complexType name="EllipsoidRefType">
  <annotation>
    <documentation>Association to an ellipsoid, either referencing or containing the definition of that ellipsoid.</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:DictionaryEntryType">
      <sequence>
        <element ref="gml:Ellipsoid" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="secondDefiningParameter" type="gml:SecondDefiningParameterType"/>
<!-- ===== -->
<complexType name="SecondDefiningParameterType">
  <annotation>
    <documentation>Definition of the second parameter that defines the shape of an ellipsoid. An ellipsoid requires two defining parameters: semi-major axis and inverse flattening or semi-major axis and semi-minor axis. When the reference body is a sphere rather than an ellipsoid, only a single defining parameter is required, namely the radius of the sphere; in that case, the semi-major axis "degenerates" into the radius of the sphere.</documentation>
  </annotation>
  <choice>
    <element ref="gml:inverseFlattening"/>
    <element ref="gml:semiMinorAxis"/>
    <element ref="gml:isSphere"/>
  </choice>
</complexType>
<!-- ===== -->
<element name="inverseFlattening" type="gml:ScaleType">

```

© OGC and ISO 2004 – All rights reserved

```

<annotation>
  <documentation>Inverse flattening value of the ellipsoid. </documentation>
</annotation>
</element>
<!-- ===== -->
<element name="semiMinorAxis" type="gml:LengthType">
  <annotation>
    <documentation>Length of the semi-minor axis of the ellipsoid. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="isSphere">
  <annotation>
    <documentation>The ellipsoid is degenerate and is actually a sphere. The sphere is completely defined by the semi-major axis, which is the radius of the sphere. </documentation>
  </annotation>
  <simpleType>
    <restriction base="string">
      <enumeration value="sphere"/>
    </restriction>
  </simpleType>
</element>
<!-- ===== -->
</schema>

```

## C.8 defaultStyle.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.4 U (http://www.xmlspy.com) by Milan Trninic (Galdos Systems Inc.) -->
<schema targetNamespace="http://www.opengis.net/gml" xmlns:smil20="http://www.w3.org/2001/SMIL20">
  xmlns:gml="http://www.opengis.net/gml" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="3.1.0">
    <annotation>
      <appinfo source="urn:opengis:specification:gml:schema-defaultStyle:v3.1.0">defaultStyle.xsd</appinfo>
      <documentation>
        <name>defaultStyle.xsd</name>
        <version>3.1.0</version>
        <scope/>
        <description>Default Style schema for GML 3.1.0</description>
        <copyright>Copyright (c) 2001-2003 OGC, All Rights Reserved.</copyright>
        <conformance>reference to ISO Specifications</conformance>
      </documentation>
    </annotation>
    <!-- ===== -->
    includes and imports
    <!-- ===== -->
    <include schemaLocation="measures.xsd"/>
    <import namespace="http://www.w3.org/2001/SMIL20" schemaLocation="../smil/smil20.xsd"/>
    <!-- ===== -->
    the Style property
    <!-- ===== -->
    <element name="defaultStyle" type="gml:DefaultStylePropertyType">
      <annotation>
        <documentation>Top-level property. Used in application schemas to "attach" the styling information to GML data. The link between the data and the style should be established through this property only.</documentation>
      </annotation>
    </element>
    <!-- ===== -->
    <complexType name="DefaultStylePropertyType">
      <annotation>
        <documentation>[complexType of] Top-level property. Used in application schemas to "attach" the styling information to GML data. The link between the data and the style should be established through this property only.</documentation>
      </annotation>
      <sequence>
        <element ref="gml:_Style" minOccurs="0"/>
      </sequence>
      <attribute name="about" type="anyURI" use="optional"/>
      <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </complexType>
    <!-- ===== -->

```

the Style

```

=====
<element name="_Style" type="gml:AbstractStyleType" abstract="true" substitutionGroup="gml:_GML">
  <annotation>
    <documentation>The value of the top-level property. It is an abstract element. Used as the head element of the substitution group for extensibility purposes.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractStyleType" abstract="true">
  <annotation>
    <documentation>[complexType of] The value of the top-level property. It is an abstract element. Used as the head element of the substitution group for extensibility purposes.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGMLType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="Style" type="gml:StyleType" substitutionGroup="gml:_Style">
  <annotation>
    <documentation>Predefined concrete value of the top-level property. Encapsulates all other styling information.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="StyleType">
  <annotation>
    <documentation>[complexType of] Predefined concrete value of the top-level property. Encapsulates all other styling information.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractStyleType">
      <sequence>
        <element ref="gml:featureStyle" maxOccurs="unbounded"/>
        <element ref="gml:graphStyle" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== --
Feature Style Property
=====
<element name="featureStyle" type="gml:FeatureStylePropertyType">
  <annotation>
    <documentation/>
  </annotation>
</element>
<!-- ===== -->
<complexType name="FeatureStylePropertyType">
  <annotation>
    <documentation/>
  </annotation>
  <sequence>
    <element ref="gml:FeatureStyle" minOccurs="0"/>
  </sequence>
  <attribute name="about" type="anyURI" use="optional"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== --
Feature Style
=====
<element name="FeatureStyle" type="gml:FeatureStyleType" substitutionGroup="gml:_GML">
  <annotation>
    <documentation>The style descriptor for features.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="FeatureStyleType">
```

```

<annotation>
  <documentation>[complexType of] The style descriptor for features.</documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractGMLType">
    <sequence>
      <element name="featureConstraint" type="string" minOccurs="0"/>
      <element ref="gml:geometryStyle" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:topologyStyle" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:labelStyle" minOccurs="0"/>
    </sequence>
    <attribute name="featureType" type="string" use="optional"/>
    <attribute name="baseType" type="string" use="optional"/>
    <attribute name="queryGrammar" type="gml:QueryGrammarEnumeration"/>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
<simpleType name="QueryGrammarEnumeration">
  <annotation>
    <documentation>Used to specify the grammar of the feature query mechanism.</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="xpath"/>
    <enumeration value="xquery"/>
    <enumeration value="other"/>
  </restriction>
</simpleType>
<!-- =====
Base style descriptor type (for geometry, topology, label, graph)
===== -->
<complexType name="BaseStyleDescriptorType">
  <annotation>
    <documentation>Base complex type for geometry, topology, label and graph styles.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <sequence>
        <element name="spatialResolution" type="gml:ScaleType" minOccurs="0"/>
        <element name="styleVariation" type="gml:StyleVariationType" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="smil20:animate" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="smil20:animateMotion" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="smil20:animateColor" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="smil20:set" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- =====
Geometry Style Property
===== -->
<element name="geometryStyle" type="gml:GeometryStylePropertyType">
  <annotation>
    <documentation/>
  </annotation>
</element>
<!-- ===== -->
<complexType name="GeometryStylePropertyType">
  <annotation>
    <documentation/>
  </annotation>
  <sequence>
    <element ref="gml:GeometryStyle" minOccurs="0"/>
  </sequence>
  <attribute name="about" type="anyURI" use="optional"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- =====
Geometry Style
===== -->
<element name="GeometryStyle" type="gml:GeometryStyleType" substitutionGroup="gml:_GML">

```

```

<annotation>
  <documentation>The style descriptor for geometries of a feature.</documentation>
</annotation>
</element>
<!-- ===== -->
<complexType name="GeometryStyleType">
  <annotation>
    <documentation>[complexType of] The style descriptor for geometries of a feature.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:BaseStyleDescriptorType">
      <sequence>
        <choice>
          <element ref="gml:symbol"/>
          <element name="style" type="string">
            <annotation>
              <appinfo>deprecated</appinfo>
              <documentation>Deprecated in GML version 3.1.0. Use symbol with inline content instead.</documentation>
            </annotation>
          </element>
        </choice>
        <element ref="gml:labelStyle" minOccurs="0"/>
      </sequence>
      <attribute name="geometryProperty" type="string"/>
      <attribute name="geometryType" type="string"/>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
Topology Style Property
===== -->
<element name="topologyStyle" type="gml:TopologyStylePropertyType">
  <annotation>
    <documentation/>
  </annotation>
</element>
<!-- ===== -->
<complexType name="TopologyStylePropertyType">
  <annotation>
    <documentation/>
  </annotation>
  <sequence>
    <element ref="gml:TopologyStyle" minOccurs="0"/>
  </sequence>
  <attribute name="about" type="anyURI" use="optional"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
Topology Style
===== -->
<element name="TopologyStyle" type="gml:TopologyStyleType" substitutionGroup="gml:_GML">
  <annotation>
    <documentation>The style descriptor for topologies of a feature. Describes individual topology elements styles.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="TopologyStyleType">
  <annotation>
    <documentation>[complexType of] The style descriptor for topologies of a feature. Describes individual topology elements styles.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:BaseStyleDescriptorType">
      <sequence>
        <choice>
          <element ref="gml:symbol"/>
          <element name="style" type="string">

```

```

        <annotation>
            <appinfo>deprecated</appinfo>
            <documentation>Deprecated in GML version 3.1.0. Use symbol with inline content
instead.</documentation>
        </annotation>
        <element>
        </choice>
        <element ref="gml:labelStyle" minOccurs="0"/>
    </sequence>
    <attribute name="topologyProperty" type="string"/>
    <attribute name="topologyType" type="string"/>
</extension>
</complexContent>
</complexType>
<!-- =====
Label Style Property
===== -->
<element name="labelStyle" type="gml:LabelStylePropertyType">
    <annotation>
        <documentation/>
    </annotation>
</element>
<!-- =====
<complexType name="LabelStylePropertyType">
    <annotation>
        <documentation/>
    </annotation>
    <sequence>
        <element ref="gml:LabelStyle" minOccurs="0"/>
    </sequence>
    <attribute name="about" type="anyURI" use="optional"/>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- =====
Label Style
===== -->
<element name="LabelStyle" type="gml:LabelStyleType" substitutionGroup="gml:_GML">
    <annotation>
        <documentation>The style descriptor for labels of a feature, geometry or topology.</documentation>
    </annotation>
</element>
<!-- =====
<complexType name="LabelStyleType">
    <annotation>
        <documentation>[complexType of] The style descriptor for labels of a feature, geometry or
topology.</documentation>
    </annotation>
    <complexContent>
        <extension base="gml:BaseStyleDescriptorType">
            <sequence>
                <element name="style" type="string"/>
                <element name="label" type="gml:LabelType"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<!-- =====
Graph Style Property
===== -->
<element name="graphStyle" type="gml:GraphStylePropertyType">
    <annotation>
        <documentation/>
    </annotation>
</element>
<!-- =====
<complexType name="GraphStylePropertyType">
    <annotation>
        <documentation/>
    </annotation>
    <sequence>
        <element ref="gml:GraphStyle" minOccurs="0"/>
    </sequence>
</complexType>

```

```

</sequence>
<attribute name="about" type="anyURI" use="optional"/>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== Graph Style ===== -->
<element name="GraphStyle" type="gml:GraphStyleType" substitutionGroup="gml:_GML">
  <annotation>
    <documentation>The style descriptor for a graph consisting of a number of features. Describes graph-specific style attributes.</documentation>
  </annotation>
</element>
<!-- ===== GraphStyleType ===== -->
<complexType name="GraphStyleType">
  <annotation>
    <documentation>[complexType of] The style descriptor for a graph consisting of a number of features. Describes graph-specific style attributes.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:BaseStyleDescriptorType">
      <sequence>
        <element name="planar" type="boolean" minOccurs="0"/>
        <element name="directed" type="boolean" minOccurs="0"/>
        <element name="grid" type="boolean" minOccurs="0"/>
        <element name="minDistance" type="double" minOccurs="0"/>
        <element name="minAngle" type="double" minOccurs="0"/>
        <element name="graphType" type="gml:GraphTypeType" minOccurs="0"/>
        <element name="drawingType" type="gml:DrawingTypeType" minOccurs="0"/>
        <element name="lineType" type="gml:LineTypeType" minOccurs="0"/>
        <element name="aestheticCriteria" type="gml:AestheticCriteriaType" minOccurs="0"
maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== Common elements ===== -->
<element name="symbol" type="gml:SymbolType">
  <annotation>
    <documentation>The symbol property. Extends the gml:AssociationType to allow for remote referencing of symbols.</documentation>
  </annotation>
</element>
<!-- ===== SymbolType ===== -->
<complexType name="SymbolType">
  <annotation>
    <documentation>[complexType of] The symbol property. Allows for remote referencing of symbols.</documentation>
  </annotation>
  <sequence>
    <any processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="symbolType" type="gml:SymbolTypeEnumeration" use="required"/>
  <attribute ref="gml:transform" use="optional"/>
  <attribute name="about" type="anyURI" use="optional"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== SymbolTypeEnumeration ===== -->
<simpleType name="SymbolTypeEnumeration">
  <annotation>
    <documentation>Used to specify the type of the symbol used.</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="svg"/>
    <enumeration value="xpath"/>
    <enumeration value="other"/>
  </restriction>
</simpleType>

```

```

<!-- ===== -->
<complexType name="LabelType" mixed="true">
  <annotation>
    <documentation>Label is mixed -- composed of text and XPath expressions used to extract the useful information from the feature.</documentation>
  </annotation>
  <sequence>
    <element name="LabelExpression" type="string" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute ref="gml:transform" use="optional"/>
</complexType>
<!-- ===== -->
<attribute name="transform" type="string">
  <annotation>
    <documentation>Defines the geometric transformation of entities. There is no particular grammar defined for this value.</documentation>
  </annotation>
</attribute>
<!-- ===== -->
<complexType name="StyleVariationType">
  <annotation>
    <documentation>Used to vary individual graphic parameters and attributes of the style, symbol or text.</documentation>
  </annotation>
  <simpleContent>
    <extension base="string">
      <attribute name="styleProperty" type="string" use="required"/>
      <attribute name="featurePropertyRange" type="string" use="optional"/>
    </extension>
  </simpleContent>
</complexType>
<!-- ===== --
Graph parameters types
===== -->
<simpleType name="GraphTypeType">
  <annotation>
    <documentation>Graph-specific styling property.</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="TREE"/>
    <enumeration value="BICONNECTED"/>
  </restriction>
</simpleType>
<!-- ===== -->
<simpleType name="DrawingTypeType">
  <annotation>
    <documentation>Graph-specific styling property.</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="POLYLINE"/>
    <enumeration value="ORTHOGONAL"/>
  </restriction>
</simpleType>
<!-- ===== -->
<simpleType name="LineTypeType">
  <annotation>
    <documentation>Graph-specific styling property.</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="STRAIGHT"/>
    <enumeration value="BENT"/>
  </restriction>
</simpleType>
<!-- ===== -->
<simpleType name="AestheticCriteriaType">
  <annotation>
    <documentation>Graph-specific styling property.</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="MIN_CROSSINGS"/>
    <enumeration value="MIN_AREA"/>
  </restriction>
</simpleType>

```

```

<enumeration value="MIN_BENDS"/>
<enumeration value="MAX_BENDS"/>
<enumeration value="UNIFORM_BENDS"/>
<enumeration value="MIN_SLOPES"/>
<enumeration value="MIN_EDGE_LENGTH"/>
<enumeration value="MAX_EDGE_LENGTH"/>
<enumeration value="UNIFORM_EDGE_LENGTH"/>
<enumeration value="MAX_ANGULAR_RESOLUTION"/>
<enumeration value="MIN_ASPECT_RATIO"/>
<enumeration value="MAX_SYMMETRIES"/>
</restriction>
</simpleType>
</schema>

```

## C.9 dictionary.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns:gml="http://www.opengis.net/gml"
       xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="3.1.0">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-xsd:dictionary:v3.1.0"/>
        <documentation>
            Dictionary schema for GML 3.1.0.
            Components to support the lists of definitions.
        </documentation>
        </annotation>
        <!-- =====
        includes and imports
        =====-->
        <include schemaLocation="gmlBase.xsd"/>
        <!-- =====
        <!-- ===== Dictionary and Definition components ==>
        <!-- =====-->
        <element name="Definition" type="gml:DefinitionType" substitutionGroup="gml:_GML"/>
        <!-- =====
        <complexType name="DefinitionType">
            <annotation>
                <documentation>A definition, which can be included in or referenced by a dictionary. In this extended type, the
                inherited "description" optional element can hold the definition whenever only text is needed. The inherited "name" elements can
                provide one or more brief terms for which this is the definition. The inherited "metaDataProperty" elements can be used to
                reference or include more information about this definition.
                The gml:id attribute is required - it must be possible to reference this definition using this handle. </documentation>
            </annotation>
            <restriction base="gml:AbstractGMLType">
                <sequence>
                    <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
                    <element ref="gml:description" minOccurs="0"/>
                    <element ref="gml:name" maxOccurs="unbounded"/>
                </sequence>
                <attribute ref="gml:id" use="required"/>
            </restriction>
        </complexType>
        <!-- =====
        <element name="Dictionary" type="gml:DictionaryType" substitutionGroup="gml:Definition"/>
        <element name="DefinitionCollection" type="gml:DictionaryType" substitutionGroup="gml:Definition"/>
        <!-- =====
        <complexType name="DictionaryType">
            <annotation>
                <documentation>A non-abstract bag that is specialized for use as a dictionary which contains a set of definitions.
                These definitions are referenced from other places, in the same and different XML documents. In this restricted type, the inherited
                optional "description" element can be used for a description of this dictionary. The inherited optional "name" element can be used
                for the name(s) of this dictionary. The inherited "metaDataProperty" elements can be used to reference or contain more
                information about this dictionary. The inherited required gml:id attribute allows the dictionary to be referenced using this handle.
                </documentation>
            </annotation>
        </complexType>
    <!-- =====
    
```

```

<complexContent>
  <extension base="gml:DefinitionType">
    <choice minOccurs="0" maxOccurs="unbounded">
      <element ref="gml:dictionaryEntry">
        <annotation>
          <documentation>An entry in this dictionary. The content of an entry can itself be a lower level
          dictionary or definition collection. This element follows the standard GML property model, so the value may be provided directly or
          by reference. Note that if the value is provided by reference, this definition does not carry a handle (gml:id) in this context, so
          does not allow external references to this specific entry in this context. When used in this way the referenced definition will usually
          be in a dictionary in the same XML document. </documentation>
        </annotation>
      </element>
      <element ref="gml:indirectEntry">
        <annotation>
          <documentation>An identified reference to a remote entry in this dictionary, to be used when this
          entry should be identified to allow external references to this specific entry. </documentation>
        </annotation>
      </element>
    </choice>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="dictionaryEntry" type="gml:DictionaryEntryType"/>
<element name="definitionMember" type="gml:DictionaryEntryType" substitutionGroup="gml:dictionaryEntry"/>
<!-- ===== -->
<complexType name="DictionaryEntryType">
  <annotation>
    <documentation>An entry in a dictionary of definitions. An instance of this type contains or refers to a definition
    object.
  </documentation>

```

The number of definitions contained in this dictionaryEntry is restricted to one, but a DefinitionCollection or Dictionary that contains multiple definitions can be substituted if needed. Specialized descendants of this dictionaryEntry might be restricted in an application schema to allow only including specified types of definitions as valid entries in a dictionary. </documentation>

```

    </annotation>
    <sequence>
      <element ref="gml:Definition" minOccurs="0">
        <annotation>
          <documentation>This element in a dictionary entry contains the actual definition. </documentation>
        </annotation>
      </element>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup">
      <annotation>
        <documentation>A non-identified reference to a remote entry in this dictionary, to be used when this entry need
        not be identified to allow external references to this specific entry. The remote entry referenced will usually be in a dictionary in
        the same XML document. This element will usually be used in dictionaries that are inside of another dictionary. </documentation>
      </annotation>
    </attributeGroup>
  </complexType>
  <!-- ===== -->
<element name="indirectEntry" type="gml:IndirectEntryType"/>
<!-- ===== -->
<complexType name="IndirectEntryType">
  <annotation>
    <documentation>An entry in a dictionary of definitions that contains a GML object which references a remote
    definition object. This entry is expected to be convenient in allowing multiple elements in one XML document to contain short
    (abbreviated XPointer) references, which are resolved to an external definition provided in a Dictionary element in the same XML
    document. Specialized descendants of this dictionaryEntry might be restricted in an application schema to allow only including
    specified types of definitions as valid entries in a dictionary. </documentation>
  </annotation>
  <sequence>
    <element ref="gml:DefinitionProxy"/>
  </sequence>
</complexType>
<!-- ===== -->
<element name="DefinitionProxy" type="gml:DefinitionProxyType" substitutionGroup="gml:Definition"/>
<!-- ===== -->
<complexType name="DefinitionProxyType">
  <annotation>

```

```

<documentation>A proxy entry in a dictionary of definitions. An element of this type contains a reference to a remote definition object. This entry is expected to be convenient in allowing multiple elements in one XML document to contain short (abbreviated XPointer) references, which are resolved to an external definition provided in a Dictionary element in the same XML document.</documentation>
</annotation>
<complexContent>
  <extension base="gml:DefinitionType">
    <sequence>
      <element ref="gml:definitionRef">
        <annotation>
          <documentation>A reference to a remote entry in this dictionary, used when this dictionary entry is identified to allow external references to this specific entry. The remote entry referenced can be in a dictionary in the same or different XML document.</documentation>
          </annotation>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="definitionRef" type="gml:ReferenceType"/>
<!-- ===== -->

```

</schema>

## C.10 direction.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns="http://www.w3.org/2001/XMLSchema" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" version="3.1.0">
  <xsd:annotation>
    <xsd:appinfo source="urn:opengis:specification:gml:schema-xsd:direction:v3.1.0">direction.xsd</xsd:appinfo>
    <xsd:documentation>
      This schema defines "direction" element and type.
    </xsd:documentation>
  </xsd:annotation>
  <!-- ===== -->
  includes and imports
  =====-->
  <xsd:include schemaLocation="geometryBasic0d1d.xsd"/>
  <!-- ===== -->
  <!-- ===== -->
  <xsd:element name="direction" type="gml:DirectionPropertyType"/>
  <!-- ===== -->
  <xsd:complexType name="DirectionPropertyType">
    <xsd:annotation>
      <xsd:documentation/>
    </xsd:annotation>
    <xsd:choice>
      <xsd:element ref="gml:DirectionVector"/>
      <xsd:element ref="gml:CompassPoint"/>
      <xsd:element name="DirectionKeyword" type="gml:CodeType"/>
      <xsd:element name="DirectionString" type="gml:StringOrRefType"/>
    </xsd:choice>
    <xsd:attributeGroup ref="gml:AssociationAttributeGroup"/>
  </xsd:complexType>
  <!-- ===== -->
  <xsd:element name="DirectionVector" type="gml:DirectionVectorType"/>
  <!-- ===== -->
  <xsd:complexType name="DirectionVectorType">
    <xsd:annotation>
      <xsd:documentation>Direction expressed as a vector, either using components, or using angles.</xsd:documentation>
    </xsd:annotation>
    <xsd:choice>
      <xsd:element ref="gml:vector"/>
      <xsd:sequence>
        <xsd:element name="horizontalAngle" type="gml:AngleType"/>
      </xsd:sequence>
    </xsd:choice>
  </xsd:complexType>

```

```

        <xsd:element name="verticalAngle" type="gml:AngleType"/>
    </xsd:sequence>
</xsd:choice>
</xsd:complexType>
<!--
<xsd:element name="CompassPoint" type="gml:CompassPointEnumeration"/>
-->
<!--
<xsd:simpleType name="CompassPointEnumeration">
    <xsd:restriction base="string">
        <xsd:enumeration value="N"/>
        <xsd:enumeration value="NNE"/>
        <xsd:enumeration value="NE"/>
        <xsd:enumeration value="ENE"/>
        <xsd:enumeration value="E"/>
        <xsd:enumeration value="ESE"/>
        <xsd:enumeration value="SE"/>
        <xsd:enumeration value="SSE"/>
        <xsd:enumeration value="S"/>
        <xsd:enumeration value="SSW"/>
        <xsd:enumeration value="SW"/>
        <xsd:enumeration value="WSW"/>
        <xsd:enumeration value="W"/>
        <xsd:enumeration value="WNW"/>
        <xsd:enumeration value="NW"/>
        <xsd:enumeration value="NNW"/>
    </xsd:restriction>
</xsd:simpleType>
-->
</xsd:schema>
```

## C.11 dynamicFeature.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
       xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" version="3.1.0">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-xsd:dynamicFeature:v3.1.0"/>
        <documentation xml:lang="en">
            Basic support for tracking moving objects and objects with changing state.
            Copyright (c) 2002 OGC, All Rights Reserved.
        </documentation>
    </annotation>
    <!-- ===== -->
    <include schemaLocation="feature.xsd"/>
    <include schemaLocation="direction.xsd"/>
    <!-- ===== -->
    <element name="dataSource" type="gml:StringOrRefType"/>
    <element name="status" type="gml:StringOrRefType"/>
    <!-- ===== -->
    <element name="_TimeSlice" type="gml:AbstractTimeSliceType" abstract="true" substitutionGroup="gml:_GML"/>
    <!-- ===== -->
    <complexType name="AbstractTimeSliceType" abstract="true">
        <annotation>
            <documentation xml:lang="en">
                A timeslice encapsulates the time-varying properties of a dynamic feature--it
                must be extended to represent a timestamped projection of a feature. The dataSource
                property describes how the temporal data was acquired.
            </documentation>
        </annotation>
        <complexContent>
            <extension base="gml:AbstractGMLType">
                <sequence>
                    <element ref="gml:validTime"/>
                    <element ref="gml:dataSource" minOccurs="0"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <!-- ===== -->
    <element name="MovingObjectStatus" type="gml:MovingObjectStatusType" substitutionGroup="gml:_TimeSlice"/>
```

```

<!-- ===== -->
<complexType name="MovingObjectStatusType">
  <annotation>
    <documentation xml:lang="en">
      This type encapsulates various dynamic properties of moving objects
      (points, lines, regions). It is useful for dealing with features whose
      geometry or topology changes over time.      </documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractTimeSliceType">
        <sequence>
          <element ref="gml:location"/>
          <element name="speed" type="gml:MeasureType" minOccurs="0"/>
          <element name="bearing" type="gml:DirectionPropertyType" minOccurs="0"/>
          <element name="acceleration" type="gml:MeasureType" minOccurs="0"/>
          <element name="elevation" type="gml:MeasureType" minOccurs="0"/>
          <element ref="gml:status" minOccurs="0"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
<!-- ===== -->
<element name="history" type="gml:HistoryPropertyType"/>
<!-- ===== -->
<complexType name="History.PropertyType">
  <annotation>
    <documentation xml:lang="en">
      The history relationship associates a feature with a sequence of TimeSlice instances.
    </documentation>
    </annotation>
    <sequence>
      <element ref="gml:_TimeSlice" maxOccurs="unbounded"/>
    </sequence>
  </complexType>
<!-- ===== -->
<element name="track" type="gml:TrackType" substitutionGroup="gml:history"/>
<!-- ===== -->
<complexType name="TrackType">
  <annotation>
    <documentation xml:lang="en">      The track of a moving object is a sequence of specialized timeslices      that
      indicate the status of the object.      </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:History.PropertyType">
      <sequence>
        <element ref="gml:MovingObjectStatus" maxOccurs="unbounded"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<group name="dynamicProperties">
  <sequence>
    <element ref="gml:validTime" minOccurs="0"/>
    <element ref="gml:history" minOccurs="0"/>
    <element ref="gml:dataSource" minOccurs="0"/>
  </sequence>
</group>
<!-- ===== -->
<complexType name="DynamicFeatureType">
  <annotation>
    <documentation>A dynamic feature may possess a history and/or a timestamp.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <group ref="gml:dynamicProperties"/>
    </extension>
  </complexContent>
</complexType>

```

```
<!-- ===== -->
<complexType name="DynamicFeatureCollectionType">
  <annotation>
    <documentation>A dynamic feature collection may possess a history and/or a timestamp.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:FeatureCollectionType">
      <group ref="gml:dynamicProperties"/>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
</schema>
```

## C.12 feature.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
       xmlns:sch="http://www.ascc.net/xml/schematron" xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified"
       version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:feature:v3.1.0"/>
    <documentation>
      GML Feature schema.
      Copyright (c) 2001-2003 OGC, All Rights Reserved.
    </documentation>
  </annotation>
  <!-- ===== includes and imports ===== -->
  <!-- ===== -->
  <!-- include schemaLocation="geometryBasic2d.xsd"/>
  <!-- include schemaLocation="temporal.xsd"/>
  <!-- ===== -->
  <element name="_Feature" type="gml:AbstractFeatureType" abstract="true" substitutionGroup="gml:_GML"/>
  <!-- ===== -->
  <complexType name="AbstractFeatureType" abstract="true">
    <annotation>
      <documentation> An abstract feature provides a set of common properties, including id, metaDataProperty, name and description inherited from AbstractGMLType, plus boundedBy. A concrete feature type must derive from this type and specify additional properties in an application schema. A feature must possess an identifying attribute ('id' - 'fid' has been deprecated). </documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractGMLType">
        <sequence>
          <element ref="gml:boundedBy" minOccurs="0"/>
          <element ref="gml:location" minOccurs="0">
            <annotation>
              <appinfo>deprecated</appinfo>
              <documentation>deprecated in GML version 3.1</documentation>
            </annotation>
          </element>
          <!-- additional properties must be specified in an application schema -->
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <!-- ===== -->
  <!-- ===== -->
  <element name="boundedBy" type="gml:BoundingShapeType"/>
  <!-- ===== -->
  <complexType name="BoundingShapeType">
    <annotation>
      <documentation>Bounding shape.</documentation>
    </annotation>
    <sequence>
      <choice>
        <element ref="gml:Envelope"/>
        <element ref="gml:Null"/>
      </choice>
    </sequence>
  </complexType>
```

```

        </sequence>
    </complexType>
<!-- ===== -->
<element name="EnvelopeWithTimePeriod" type="gml:EnvelopeWithTimePeriodType" substitutionGroup="gml:Envelope"/>
<!-- ===== -->
<complexType name="EnvelopeWithTimePeriodType">
    <annotation>
        <documentation>Envelope that includes also a temporal extent.</documentation>
    </annotation>
    <complexContent>
        <extension base="gml:EnvelopeType">
            <sequence>
                <element ref="gml:timePosition" minOccurs="2" maxOccurs="2"/>
            </sequence>
                <attribute name="frame" type="anyURI" use="optional" default="#ISO-8601"/>
        </extension>
    </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== property for feature association ===== -->
<element name="featureMember" type="gml:FeaturePropertyType"/>
<element name="featureProperty" type="gml:FeaturePropertyType"/>
<!-- ===== -->
<complexType name="Feature.PropertyType">
    <annotation>
        <documentation>Container for a feature - follow gml:AssociationType pattern.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:_Feature" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- ===== property for association of an array of features ===== -->
<element name="featureMembers" type="gml:FeatureArray.PropertyType"/>
<!-- ===== -->
<complexType name="FeatureArray.PropertyType">
    <annotation>
        <documentation>Container for features - follow gml:ArrayAssociationType pattern.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:_Feature" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
</complexType>
<!-- ===== -->
<!-- ===== -->
<!-- ===== _FeatureCollection type="gml:AbstractFeatureCollectionType" abstract="true" substitutionGroup="gml:_Feature"/>
<!-- ===== -->
<complexType name="AbstractFeatureCollectionType" abstract="true">
    <annotation>
        <documentation>A feature collection contains zero or more features. </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element ref="gml:featureMember" minOccurs="0" maxOccurs="unbounded"/>
                <element ref="gml:featureMembers" minOccurs="0"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="FeatureCollection" type="gml:FeatureCollectionType" substitutionGroup="gml:_Feature"/>
<!-- ===== -->
<complexType name="FeatureCollectionType">
    <annotation>
        <documentation>Concrete generic feature collection. </documentation>
    </annotation>
</complexType>

```

```

<complexContent>
  <extension base="gml:AbstractFeatureCollectionType"/>
</complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="LocationKeyWord" type="gml:CodeType"/>
<element name="LocationString" type="gml:StringOrRefType"/>
<!-- ===== -->
<!-- ===== common aliases for geometry properties ===== -->
<element name="centerOf" type="gml:PointPropertyType"/>
<element name="position" type="gml:PointPropertyType"/>
<element name="edgeOf" type="gml:CurvePropertyType"/>
<element name="centerLineOf" type="gml:CurvePropertyType"/>
<element name="extentOf" type="gml:SurfacePropertyType"/>
<!-- ===== -->
<!-- ===== deprecated components ===== -->
<complexType name="BoundedFeatureType" abstract="true">
  <annotation>
    <documentation> Makes boundedBy mandatory</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:AbstractFeatureType">
      <sequence>
        <group ref="gml:StandardObjectProperties"/>
        <element ref="gml:boundedBy"/>
        <element ref="gml:location" minOccurs="0">
          <annotation>
            <appinfo>deprecated</appinfo>
            <documentation>deprecated in GML version 3.1</documentation>
          </annotation>
        </element>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="location" type="gml:LocationPropertyType">
  <annotation>
    <documentation>Deprecated in GML 3.1.0</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="Location.PropertyType">
  <annotation>
    <documentation>Convenience property for generalised location.

```

```

        </annotation>
    <complexContent>
        <extension base="gml:LocationPropertyType">
            <attribute name="priority" type="string" use="optional"/>
        </extension>
    </complexContent>
</complexType>
<!-- ===== -->
</schema>

```

### C.13 geometryAggregates.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:sch="http://www.ascc.net/xml/schematron" xmlns:gml="http://www.opengis.net/gml"
    xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDefault="qualified" version="3.1.0">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-
xsd:geometryAggregates:v3.1.0">geometryAggregates.xsd</appinfo>
        <documentation>
            Copyright (c) 2001-2002 OGC, All Rights Reserved.
        </documentation>
        </annotation>
        <include schemaLocation="geometryPrimitives.xsd"/>
    <!-- ===== -->
    <!-- aggregate geometry objects -->
    <!-- ===== -->
    <!-- ===== -->
    <element name="_GeometricAggregate" type="gml:AbstractGeometricAggregateType" abstract="true"
        substitutionGroup="gml:_Geometry">
        <annotation>
            <documentation>The "_GeometricAggregate" element is the abstract head of the substitution group for all geometric
            aggregates.</documentation>
        </annotation>
        </element>
    <!-- ===== -->
    <complexType name="AbstractGeometricAggregateType" abstract="true">
        <annotation>
            <documentation>This is the abstract root type of the geometric aggregates.</documentation>
        </annotation>
        <complexContent>
            <extension base="gml:AbstractGeometryType"/>
        </complexContent>
    </complexType>
    <!-- ===== -->
    <element name="MultiGeometry" type="gml:MultiGeometryType" substitutionGroup="gml:_GeometricAggregate"/>
    <!-- ===== -->
    <complexType name="MultiGeometryType">
        <annotation>
            <documentation>
                A geometry collection must include one or more geometries, referenced through geometryMember elements.
            </documentation>
        </annotation>
        <complexContent>
            <extension base="gml:AbstractGeometricAggregateType">
                <sequence>
                    <annotation>
                        <documentation>The members of the geometric aggregate can be specified either using the "standard"
                        property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection.
                        NOTE: Array properties cannot reference remote geometry elements.</documentation>
                    </annotation>
                    <element ref="gml:geometryMember" minOccurs="0" maxOccurs="unbounded"/>
                    <element ref="gml:geometryMembers" minOccurs="0"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <!-- ===== -->

```

```

<element name="multiGeometryProperty" type="gml:MultiGeometryPropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:multiGeometryProperty">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
    <documentation>This property element either references a geometric aggregate via the XLink-attributes or contains the "multi geometry" element. multiGeometryProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for _GeometricAggregate.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="MultiGeometryPropertyType">
  <annotation>
    <documentation>A property that has a geometric aggregate as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>
  </annotation>
  <sequence>
    <element ref="gml:_GeometricAggregate" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup">
    <annotation>
      <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties.


A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the gml:AssociationAttributeGroup.</documentation>
    </annotation>
  </attributeGroup>
</complexType>
<!-- ===== -->
<element name="MultiPoint" type="gml:MultiPointType" substitutionGroup="gml:_GeometricAggregate"/>
<!-- ===== -->
<complexType name="MultiPointType">
  <annotation>
    <documentation>A MultiPoint is defined by one or more Points, referenced through pointMember elements.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGeometricAggregateType">
      <sequence>
        <annotation>
          <documentation>The members of the geometric aggregate can be specified either using the "standard" property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection. NOTE: Array properties cannot reference remote geometry elements.</documentation>
        </annotation>
        <element ref="gml:pointMember" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:pointMembers" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="multiPointProperty" type="gml:MultiPointPropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:multiGeometryProperty">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>


```

```

<documentation>This property element either references a point aggregate via the XLink-attributes or contains the "multi point" element. multiPointProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for MultiPoint.</documentation>
</annotation>
</element>
<!-- ===== -->
<complexType name="MultiPointPropertyType">
  <annotation>
    <documentation>A property that has a collection of points as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>
  </annotation>
  <sequence>
    <element ref="gml:MultiPoint" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup">
    <annotation>
      <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties. A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the gml:AssociationAttributeGroup.</documentation>
    </annotation>
    </attributeGroup>
  </complexType>
<!-- ===== -->
<element name="MultiCurve" type="gml:MultiCurveType" substitutionGroup="gml:_GeometricAggregate"/>
<!-- ===== -->
<complexType name="MultiCurveType">
  <annotation>
    <documentation>A MultiCurve is defined by one or more Curves, referenced through curveMember elements.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGeometricAggregateType">
      <sequence>
        <annotation>
          <documentation>The members of the geometric aggregate can be specified either using the "standard" property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection. NOTE: Array properties cannot reference remote geometry elements.</documentation>
        </annotation>
        <element ref="gml:curveMember" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:curveMembers" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="multiCurveProperty" type="gml:MultiCurvePropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:multiCurveProperty">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
    <documentation>This property element either references a curve aggregate via the XLink-attributes or contains the "multi curve" element. multiCurveProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for MultiCurve.</documentation>
    </annotation>
  </element>
<!-- ===== -->
<complexType name="MultiCurvePropertyType">
  <annotation>

```

<documentation>A property that has a collection of curves as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>

```
</annotation>
<sequence>
  <element ref="gml:MultiCurve" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup">
  <annotation>
```

<documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties.

A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the gml:AssociationAttributeGroup.

```
  </documentation>
  </annotation>
</attributeGroup>
</complexType>
<!-- ===== -->
<element name="MultiSurface" type="gml:MultiSurfaceType" substitutionGroup="gml:_GeometricAggregate"/>
<!-- ===== -->
```

```
<complexType name="MultiSurfaceType">
  <annotation>
    <documentation>
```

A MultiSurface is defined by one or more Surfaces, referenced through surfaceMember elements.

```
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGeometricAggregateType">
      <sequence>
        <annotation>
```

<documentation>The members of the geometric aggregate can be specified either using the "standard" property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection.

NOTE: Array properties cannot reference remote geometry elements.</documentation>

```
      </annotation>
      <element ref="gml:surfaceMember" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:surfaceMembers" minOccurs="0"/>
    </sequence>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
```

```
<element name="multiSurfaceProperty" type="gml:MultiSurfacePropertyType">
  <annotation>
    <appinfo>
```

```
      <sch:pattern>
        <sch:rule context="gml:multiSurfaceProperty">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
```

<documentation>This property element either references a surface aggregate via the XLink-attributes or contains the "multi surface" element. multiSurfaceProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for MultiSurface.</documentation>

```
    </annotation>
  </element>
<!-- ===== -->
<complexType name="MultiSurfacePropertyType">
  <annotation>
```

<documentation>A property that has a collection of surfaces as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>

```
  </annotation>
  <sequence>
    <element ref="gml:MultiSurface" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup">
```

```

<annotation>
  <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties.
```

A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the `gml:AssociationAttributeGroup`.

```

    </documentation>
  </annotation>
</attributeGroup>
</complexType>
<!-- ===== -->
<element name="MultiSolid" type="gml:MultiSolidType" substitutionGroup="gml:_GeometricAggregate"/>
<!-- ===== -->
<complexType name="MultiSolidType">
  <annotation>
    <documentation>
      A MultiSolid is defined by one or more Solids, referenced through solidMember elements.
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGeometricAggregateType">
      <sequence>
        <annotation>
          <documentation>The members of the geometric aggregate can be specified either using the "standard" property or the array property style. It is also valid to use both the "standard" and the array property style in the same collection.
```

NOTE: Array properties cannot reference remote geometry elements.</documentation>

```

          </annotation>
          <element ref="gml:solidMember" minOccurs="0" maxOccurs="unbounded"/>
          <element ref="gml:solidMembers" minOccurs="0"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
<!-- ===== -->
<element name="multiSolidProperty" type="gml:MultiSolid.PropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:multiSolidProperty">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
    <documentation>This property element either references a solid aggregate via the XLink-attributes or contains the "multi solid" element. multiSolidProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for MultiSolid.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="MultiSolid.PropertyType">
  <annotation>
    <documentation>A property that has a collection of solids as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>
  </annotation>
  <sequence>
    <element ref="gml:MultiSolid" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup">
    <annotation>
      <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties.
```

A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the `gml:AssociationAttributeGroup`.

```

    </documentation>
    </annotation>
    </attributeGroup>
</complexType>
<!-- ===== -->
<!--

```

The following types and elements are deprecated and should not be used !

For backward compatibility with GML2 only

```

-->
<!-- ===== -->
<element name="MultiPolygon" type="gml:MultiPolygonType" substitutionGroup="gml:_GeometricAggregate">
    <annotation>
        <documentation>Deprecated with GML 3.0 and included for backwards compatibility with GML 2. Use the "MultiSurface" element instead.</documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="MultiLineString" type="gml:MultiLineStringType" substitutionGroup="gml:_GeometricAggregate">
    <annotation>
        <documentation>Deprecated with GML 3.0 and included for backwards compatibility with GML 2. Use the "MultiCurve" element instead.</documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="MultiLineStringType">
    <annotation>
        <documentation>
            A MultiLineString is defined by one or more LineStrings, referenced through lineStringMember elements. Deprecated with GML version 3.0. Use MultiCurveType instead.

```

`</documentation>`

```

        </annotation>
        <complexContent>
            <extension base="gml:AbstractGeometricAggregateType">
                <sequence>
                    <element ref="gml:lineStringMember" minOccurs="0" maxOccurs="unbounded"/>
                </sequence>
            </extension>
        </complexContent>
</complexType>
<!-- ===== -->
<complexType name="MultiLineStringPropertyType">
    <annotation>
        <documentation>This type is deprecated with GML 3 and shall not be used. It is included for backwards compatibility with GML 2. Use MultiCurvePropertyType instead.

```

A property that has a collection of line strings as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor

none.

```

</documentation>
</annotation>
<sequence>
    <element ref="gml:MultiLineString" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup">
    <annotation>

```

`<documentation>This attribute group includes the XLink attributes (see xlinks.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties.`

A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the `gml:AssociationAttributeGroup`.

```

        </documentation>
        </annotation>
        </attributeGroup>
</complexType>
<!-- ===== -->
<complexType name="MultiPolygonType">
```

```

<annotation>
  <documentation>
    A MultiPolygon is defined by one or more Polygons, referenced through polygonMember elements. Deprecated
    with GML version 3.0. Use MultiSurfaceType instead.
  </documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractGeometricAggregateType">
    <sequence>
      <element ref="gml:polygonMember" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
<complexType name="MultiPolygonPropertyType">
  <annotation>
    <documentation>This type is deprecated with GML 3 and shall not be used. It is included for backwards compatibility
    with GML 2. Use MultiSurfacePropertyType instead.
  </documentation>

```

A property that has a collection of polygons as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.

```

<annotation>
  <sequence>
    <element ref="gml:MultiPolygon" minOccurs="0"/>
  </sequence>
<attributeGroup ref="gml:AssociationAttributeGroup">
  <annotation>
    <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to
    reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by
    including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the
    World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links
    between resources; such links can be used to reference remote properties.
  </documentation>

```

A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the gml:AssociationAttributeGroup.

```

    </annotation>
  </attributeGroup>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="geometryMember" type="gml:Geometry.PropertyType">
  <annotation>
    <documentation>This property element either references a geometry element via the XLink-attributes or contains the
    geometry element.</documentation>
  </annotation>
</element>
<element name="geometryMembers" type="gml:GeometryArray.PropertyType">
  <annotation>
    <documentation>This property element contains a list of geometry elements. The order of the elements is significant
    and shall be preserved when processing the array.</documentation>
  </annotation>
</element>
<element name="pointMember" type="gml:Point.PropertyType">
  <annotation>
    <documentation>This property element either references a Point via the XLink-attributes or contains the Point
    element.</documentation>
  </annotation>
</element>
<element name="pointMembers" type="gml:PointArray.PropertyType">
  <annotation>
    <documentation>This property element contains a list of points. The order of the elements is significant and shall be
    preserved when processing the array.</documentation>
  </annotation>
</element>
<element name="curveMembers" type="gml:CurveArray.PropertyType">
  <annotation>

```

```

<documentation>This property element contains a list of curves. The order of the elements is significant and shall be
preserved when processing the array.</documentation>
</annotation>
</element>
<element name="surfaceMember" type="gml:SurfacePropertyType">
  <annotation>
    <documentation>This property element either references a surface via the XLink-attributes or contains the surface
element. A surface element is any element which is substitutable for "_Surface".</documentation>
  </annotation>
</element>
<element name="surfaceMembers" type="gml:SurfaceArrayPropertyType">
  <annotation>
    <documentation>This property element contains a list of surfaces. The order of the elements is significant and shall
be preserved when processing the array.</documentation>
  </annotation>
</element>
<element name="solidMember" type="gml:SolidPropertyType">
  <annotation>
    <documentation>This property element either references a solid via the XLink-attributes or contains the solid
element. A solid element is any element which is substitutable for "_Solid".</documentation>
  </annotation>
</element>
<element name="solidMembers" type="gml:SolidArrayPropertyType">
  <annotation>
    <documentation>This property element contains a list of solids. The order of the elements is significant and shall be
preserved when processing the array.</documentation>
  </annotation>
</element>
<!-- some named geometry properties - for backward compatibility with GML2 -->
<element name="multiCenterOf" type="gml:MultiPointPropertyType"/>
<element name="multiPosition" type="gml:MultiPointPropertyType"/>
<element name="multiCenterLineOf" type="gml:MultiCurvePropertyType"/>
<element name="multiEdgeOf" type="gml:MultiCurvePropertyType"/>
<element name="multiCoverage" type="gml:MultiSurfacePropertyType"/>
<element name="multiExtentOf" type="gml:MultiSurfacePropertyType"/>
<!--

```

The following types and elements are deprecated and should not be used !

```

-->
<element name="multiLocation" type="gml:MultiPointPropertyType">
  <annotation>
    <appinfo>deprecated</appinfo>
    <documentation>Deprecated with GML 3.0 and included only for backwards compatibility with GML 2.0. Use
"curveMember" instead.
This property element either references a line string via the XLink-attributes or contains the line string element.</documentation>
  </annotation>
</element>
<element name="lineStringMember" type="gml:LineStringPropertyType">
  <annotation>
    <appinfo>deprecated</appinfo>
    <documentation>Deprecated with GML 3.0 and included only for backwards compatibility with GML 2.0. Use
"curveMember" instead.
This property element either references a line string via the XLink-attributes or contains the line string element.</documentation>
  </annotation>
</element>
<element name="polygonMember" type="gml:PolygonPropertyType">
  <annotation>
    <appinfo>deprecated</appinfo>
    <documentation>Deprecated with GML 3.0 and included only for backwards compatibility with GML 2.0. Use
"surfaceMember" instead.
This property element either references a polygon via the XLink-attributes or contains the polygon element.</documentation>
  </annotation>
</element>
<!-- ===== -->
</schema>
```

## C.14 geometryBasic0d1d.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<!-- edited with XMLSPY v2004 rel. 2 U (http://www.xmlspy.com) by David Burggraf (Galdos Systems Inc) -->
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
       xmlns:sch="http://www.w3.org/2001/XMLSchema#"
       xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDefault="qualified" version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:geometryBasic0d1d:v3.1.0">geometryBasic0d1d.xsd</appinfo>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:geometryBasic0d1d:v3.1.0">
      <sch:title>Schematron validation</sch:title>
      <sch:pattern name="Check">
        <sch:rule abstract="true" id="CRSLabeleds">
          <sch:report test="not(@srsDimension) or @srsName">
            The presence of a dimension attribute implies the presence of the srsName attribute.</sch:report>
            <sch:report test="not(@axisLabels) or @srsName">
              The presence of an axisLabels attribute implies the presence of the srsName attribute.</sch:report>
              <sch:report test="not(@uomLabels) or @srsName">
                The presence of an uomLabels attribute implies the presence of the srsName attribute.</sch:report>
                <sch:report test="(not(@uomLabels) and not(@axisLabels)) or (@uomLabels and @axisLabels)">
                  The presence of an uomLabels attribute implies the presence of the axisLabels attribute and vice versa.</sch:report>
                  </sch:rule>
                </sch:pattern>
                <sch:pattern name="Count">
                  <sch:rule abstract="true" id="Count">
                    <sch:report test="not(@count) or @srsDimension">
                      The presence of a count attribute implies the presence of the dimension attribute.</sch:report>
                      </sch:rule>
                    </sch:pattern>
                  </appinfo>
                  <documentation>
                    Copyright (c) 2001-2003 OGC, All Rights Reserved.
                  </documentation>
                </annotation>
                </appinfo>
                <include schemaLocation="measures.xsd">
                  <annotation>
                    <documentation>This includes not only measures.xsd, but also units.xsd, gmlBase.xsd and basicTypes.xsd.</documentation>
                  </annotation>
                </include>
                <!-- ===== abstract supertype for geometry objects -->
                <!-- ===== -->
                <!-- ===== -->
                <element name="_Geometry" type="gml:AbstractGeometryType" abstract="true" substitutionGroup="gml:_GML">
                  <annotation>
                    <documentation>The "_Geometry" element is the abstract head of the substitution group for all geometry elements of GML 3. This includes pre-defined and user-defined geometry elements. Any geometry element must be a direct or indirect extension/restriction of AbstractGeometryType and must be directly or indirectly in the substitution group of "_Geometry".</documentation>
                  <appinfo>
                    <sch:pattern>
                      <sch:rule context="gml:_Geometry">
                        <sch:extends rule="CRSLabeleds"/>
                      </sch:rule>
                    </sch:pattern>
                  </appinfo>
                </annotation>
              </element>
              <!-- ===== -->
              <complexType name="Geometry.PropertyType">
                <annotation>
                  <documentation>A geometric property can either be any geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Note that either the reference or the contained element must be given, but not both or none.</documentation>
                  <annotation>
                    <sequence>
                      <element ref="gml:_Geometry" minOccurs="0"/>
                    </sequence>
                  <attributeGroup ref="gml:AssociationAttributeGroup">
                    <annotation>

```

<documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties. A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the `gml:AssociationAttributeGroup`.</documentation>

```

    </annotation>
  </attributeGroup>
</complexType>
<!-- ===== -->
<complexType name="GeometryArrayPropertyType">
  <annotation>
    <documentation>A container for an array of geometry elements. The elements are always contained in the array property, referencing geometry elements or arrays of geometry elements is not supported.</documentation>
  </annotation>
  <sequence>
    <element ref="gml:_Geometry" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<!-- ===== -->
<complexType name="AbstractGeometryType" abstract="true">
  <annotation>
    <documentation>All geometry elements are derived directly or indirectly from this abstract supertype. A geometry element may have an identifying attribute ("gml:id"), a name (attribute "name") and a description (attribute "description"). It may be associated with a spatial reference system (attribute "srsName"). The following rules shall be adhered: - Every geometry type shall derive from this abstract type. - Every geometry element (i.e. an element of a geometry type) shall be directly or indirectly in the substitution group of _Geometry.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <attribute name="gid" type="string" use="optional">
        <annotation>
          <documentation>This attribute is included for backward compatibility with GML 2 and is deprecated with GML 3. This identifier is superceded by "gml:id" inherited from AbstractGMLType. The attribute "gid" should not be used anymore and may be deleted in future versions of GML without further notice.</documentation>
        </annotation>
        <attribute>
          <attributeGroup ref="gml:SRSReferenceGroup"/>
        </attribute>
      </extension>
    </complexContent>
  </complexType>
<!-- ===== -->
<attributeGroup name="SRSReferenceGroup">
  <annotation>
    <documentation>Optional reference to the CRS used by this geometry, with optional additional information to simplify use when a more complete definition of the CRS is not needed. </documentation>
  </annotation>
  <attribute name="srsName" type="anyURI" use="optional">
    <annotation>
      <documentation>In general this reference points to a CRS instance of gml:CoordinateReferenceSystemType (see coordinateReferenceSystems.xsd). For well known references it is not required that the CRS description exists at the location the URI points to. If no srsName attribute is given, the CRS must be specified as part of the larger context this geometry element is part of, e.g. a geometric element like point, curve, etc. It is expected that this attribute will be specified at the direct position level only in rare cases.</documentation>
    </annotation>
  </attribute>
  <attribute name="srsDimension" type="positiveInteger" use="optional">
    <annotation>
      <documentation>The "srsDimension" is the length of coordinate sequence (the number of entries in the list). This dimension is specified by the coordinate reference system. When the srsName attribute is omitted, this attribute shall be omitted.</documentation>
    </annotation>
  </attributeGroup>
<!-- ===== -->
<attributeGroup name="SRSInformationGroup">
  <annotation>
    <documentation>Optional additional and redundant information for a CRS to simplify use when a more complete definition of the CRS is not needed. This information shall be the same as included in the more complete definition of the CRS,</documentation>
  </annotation>
</attributeGroup>

```

referenced by the srsName attribute. When the srsName attribute is included, either both or neither of the axisLabels and uomLabels attributes shall be included. When the srsName attribute is omitted, both of these attributes shall be omitted.

```
</documentation>
</annotation>
<attribute name="axisLabels" type="gml:NCNameList" use="optional">
    <annotation>
        <documentation>Ordered list of labels for all the axes of this CRS. The gml:axisAbbrev value should be used for these axis labels, after spaces and forbidden characters are removed. When the srsName attribute is included, this attribute is optional. When the srsName attribute is omitted, this attribute shall also be omitted. </documentation>
    </annotation>
</attribute>
<attribute name="uomLabels" type="gml:NCNameList" use="optional">
    <annotation>
        <documentation>Ordered list of unit of measure (uom) labels for all the axes of this CRS. The value of the string in the gml:catalogSymbol should be used for this uom labels, after spaces and forbidden characters are removed. When the axisLabels attribute is included, this attribute shall also be included. When the axisLabels attribute is omitted, this attribute shall also be omitted. </documentation>
    </annotation>
</attribute>
</attributeGroup>
<!-- ===== -->
<element name="_GeometricPrimitive" type="gml:AbstractGeometricPrimitiveType" abstract="true" substitutionGroup="gml:_Geometry">
    <annotation>
        <documentation>The "_GeometricPrimitive" element is the abstract head of the substitution group for all (pre- and user-defined) geometric primitives.</documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractGeometricPrimitiveType" abstract="true">
    <annotation>
        <documentation>This is the abstract root type of the geometric primitives. A geometric primitive is a geometric object that is not decomposed further into other primitives in the system. All primitives are oriented in the direction implied by the sequence of their coordinate tuples.</documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractGeometryType"/>
    </complexContent>
</complexType>
<!-- ===== -->
<complexType name="GeometricPrimitivePropertyType">
    <annotation>
        <documentation>A property that has a geometric primitive as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:_GeometricPrimitive" minOccurs="0"/>
    </sequence>
</complexType>
<attributeGroup ref="gml:AssociationAttributeGroup">
    <annotation>
        <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties. A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the gml:AssociationAttributeGroup. </documentation>
    </annotation>
</attributeGroup>
</complexType>
<!-- primitive geometry objects (0-dimensional) -->
<!-- ===== -->
<element name="Point" type="gml:PointType" substitutionGroup="gml:_GeometricPrimitive"/>
<!-- ===== -->
<complexType name="PointType">
    <annotation>
```

```

<documentation>A Point is defined by a single coordinate tuple.</documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractGeometricPrimitiveType">
    <sequence>
      <choice>
        <annotation>
          <documentation>GML supports two different ways to specify the direct position of a point. 1. The "pos" element is of type DirectPositionType.</documentation>
        <annotation>
          <element ref="gml:pos"/>
          <element ref="gml:coordinates">
            <annotation>
              <documentation>Deprecated with GML version 3.1.0 for coordinates with ordinate values that are numbers. Use "pos" instead. The "coordinates" element shall only be used for coordinates with ordinates that require a string representation, e.g. DMS representations.</documentation>
            </annotation>
          </element>
          <element ref="gml:coord">
            <annotation>
              <documentation>Deprecated with GML version 3.0. Use "pos" instead. The "coord" element is included for backwards compatibility with GML 2.</documentation>
            </annotation>
          </element>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="pointProperty" type="gml:PointPropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:pointProperty">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
    <documentation>This property element either references a point via the XLink-attributes or contains the point element. pointProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for Point.</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="pointRep" type="gml:PointPropertyType">
  <annotation>
    <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included for backwards compatibility with GML 3.0.0.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="PointPropertyType">
  <annotation>
    <documentation>A property that has a point as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>
    </annotation>
    <sequence>
      <element ref="gml:Point" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup">
      <annotation>
        <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties. A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the gml:AssociationAttributeGroup.</documentation>
      </annotation>
    </attributeGroup>
  </complexType>

```

```

        </annotation>
    </attributeGroup>
</complexType>
<!-- ===== -->
<element name="pointArrayProperty" type="gml:PointArrayPropertyType"/>
<!-- ===== -->
<complexType name="PointArrayPropertyType">
    <annotation>
        <documentation>A container for an array of points. The elements are always contained in the array property, referencing geometry elements or arrays of geometry elements is not supported.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:Point" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
</complexType>
<!-- ===== -->
<!-- primitive geometry objects (1-dimensional) -->
<!-- ===== -->
<element name="_Curve" type="gml:AbstractCurveType" abstract="true" substitutionGroup="gml:_GeometricPrimitive">
    <annotation>
        <documentation>The "_Curve" element is the abstract head of the substitution group for all (continuous) curve elements.</documentation>
    </annotation>
    <element>
        <annotation>
            <documentation>An abstraction of a curve to support the different levels of complexity. The curve can always be viewed as a geometric primitive, i.e. is continuous.</documentation>
        </annotation>
        <complexContent>
            <extension base="gml:AbstractGeometricPrimitiveType"/>
        </complexContent>
    </element>
<!-- ===== -->
<complexType name="AbstractCurveType" abstract="true">
    <annotation>
        <documentation>An abstraction of a curve to support the different levels of complexity. The curve can always be viewed as a geometric primitive, i.e. is continuous.</documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractGeometricPrimitiveType"/>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="curveProperty" type="gml:CurvePropertyType">
    <annotation>
        <appinfo>
            <sch:pattern>
                <sch:rule context="gml:curveProperty">
                    <sch:extends rule="hrefOrContent"/>
                </sch:rule>
            </sch:pattern>
        </appinfo>
        <documentation>This property element either references a curve via the XLink-attributes or contains the curve element. curveProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for _Curve.</documentation>
    </annotation>
    <element>
        <annotation>
            <documentation>A property that has a curve as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>
        </annotation>
        <sequence>
            <element ref="gml:_Curve" minOccurs="0"/>
        </sequence>
        <attributeGroup ref="gml:AssociationAttributeGroup">
            <annotation>
                <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties. A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the gml:AssociationAttributeGroup.</documentation>
            </annotation>
        </attributeGroup>
    </element>
</annotation>

```

```

        </annotation>
    </attributeGroup>
</complexType>
<!-- ===== -->
<element name="curveArrayProperty" type="gml:CurveArrayPropertyType"/>
<!-- ===== -->
<complexType name="CurveArrayPropertyType">
    <annotation>
        <documentation>A container for an array of curves. The elements are always contained in the array property, referencing geometry elements or arrays of geometry elements is not supported.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:_Curve" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
</complexType>
<!-- ===== -->
<element name="LineString" type="gml:LineStringType" substitutionGroup="gml:_Curve"/>
<!-- ===== -->
<complexType name="LineStringType">
    <annotation>
        <documentation>A LineString is a special curve that consists of a single segment with linear interpolation. It is defined by two or more coordinate tuples, with linear interpolation between them. It is backwards compatible with the LineString of GML 2, GM_LineString of ISO 19107 is implemented by LineStringSegment.</documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractCurveType">
            <sequence>
                <choice>
                    <annotation>
                        <documentation>GML supports two different ways to specify the control points of a line string. 1. A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve (reuse of existing points). 2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve only. The number of direct positions in the list must be at least two.</documentation>
                    </annotation>
                    <choice minOccurs="2" maxOccurs="unbounded">
                        <element ref="gml:pos"/>
                        <element ref="gml:pointProperty"/>
                        <element ref="gml:pointRep">
                            <annotation>
                                <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included for backwards compatibility with GML 3.0.0.</documentation>
                            </annotation>
                        </element>
                        <element ref="gml:coord">
                            <annotation>
                                <documentation>Deprecated with GML version 3.0. Use "pos" instead. The "coord" element is included for backwards compatibility with GML 2.</documentation>
                            </annotation>
                        </element>
                    </choice>
                    <element ref="gml:posList"/>
                    <element ref="gml:coordinates">
                        <annotation>
                            <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
                        </annotation>
                    </element>
                </choice>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<!-- positions -->
<!-- ===== -->
<element name="pos" type="gml:DirectPositionType">
    <annotation>
        <appinfo>
            <sch:pattern>
                <sch:rule context="gml:pos">

```

```

        <sch:extends rule="CRSLabeled"/>
    </sch:rule>
    </sch:pattern>
</appinfo>
</annotation>
</element>
<!-- ===== -->
<complexType name="DirectPositionType">
    <annotation>
        <documentation>DirectPosition instances hold the coordinates for a position within some coordinate reference system (CRS). Since DirectPositions, as data types, will often be included in larger objects (such as geometry elements) that have references to CRS, the "srsName" attribute will in general be missing, if this particular DirectPosition is included in a larger element with such a reference to a CRS. In this case, the CRS is implicitly assumed to take on the value of the containing object's CRS.</documentation>
    </annotation>
    <simpleContent>
        <extension base="gml:doubleList">
            <attributeGroup ref="gml:SRSReferenceGroup"/>
        </extension>
    </simpleContent>
</complexType>
<!-- ===== -->
<element name="posList" type="gml:DirectPositionListType">
    <annotation>
        <appinfo>
            <sch:pattern>
                <sch:rule context="gml:posList">
                    <sch:extends rule="CRSLabeled"/>
                </sch:rule>
            </sch:pattern>
        </appinfo>
        <appinfo>
            <sch:pattern>
                <sch:rule context="gml:posList">
                    <sch:extends rule="Count"/>
                </sch:rule>
            </sch:pattern>
        </appinfo>
    </annotation>
</element>
<!-- ===== -->
<complexType name="DirectPositionListType">
    <annotation>
        <documentation>DirectPositionList instances hold the coordinates for a sequence of direct positions within the same coordinate reference system (CRS).</documentation>
    </annotation>
    <simpleContent>
        <extension base="gml:doubleList">
            <attributeGroup ref="gml:SRSReferenceGroup"/>
            <attribute name="count" type="positiveInteger" use="optional">
                <annotation>
                    <documentation>"count" allows to specify the number of direct positions in the list. If the attribute "count" is present then the attribute "srsDimension" shall be present, too.</documentation>
                </annotation>
            </attribute>
        </extension>
    </simpleContent>
</complexType>
<!-- ===== -->
<element name="vector" type="gml:VectorType">
    <annotation>
        <appinfo>
            <sch:pattern>
                <sch:rule context="gml:vector">
                    <sch:extends rule="CRSLabeled"/>
                </sch:rule>
            </sch:pattern>
        </appinfo>
    </annotation>

```

```

</element>
<!-- ===== -->
<complexType name="VectorType">
  <annotation>
    <documentation>A Vector is an ordered set of numbers called coordinates that represent a position in a coordinate reference system (CRS). For some application the components of the position may be adjusted to yield a unit vector.</documentation>
  </annotation>
  <simpleContent>
    <restriction base="gml:DirectPositionType"/>
  </simpleContent>
</complexType>
<!-- ===== -->
<group name="geometricPositionGroup">
  <annotation>
    <documentation>A geometric position represented either by a DirectPosition or a Point.</documentation>
  </annotation>
  <choice>
    <element ref="gml:pos"/>
    <element ref="gml:pointProperty"/>
  </choice>
</group>
<!-- ===== -->
<group name="geometricPositionListGroup">
  <annotation>
    <documentation>A list of geometric positions represented either by a DirectPosition or a Point.</documentation>
  </annotation>
  <choice>
    <element ref="gml:posList"/>
    <group ref="gml:geometricPositionGroup" maxOccurs="unbounded"/>
  </choice>
</group>
<!-- ===== -->
<element name="coordinates" type="gml:CoordinatesType">
  <annotation>
    <documentation>Deprecated with GML version 3.1.0.</documentation>
  </annotation>
</element>
<!-- ===== -->
<!-- Envelope -->
<!-- ===== -->
<element name="Envelope" type="gml:EnvelopeType" substitutionGroup="gml:_Geometry"/>
<!-- ===== -->
<complexType name="EnvelopeType">
  <annotation>
    <documentation>Envelope defines an extent using a pair of positions defining opposite corners in arbitrary dimensions. The first direct position is the "lower corner" (a coordinate position consisting of all the minimal ordinates for each dimension for all points within the envelope), the second one the "upper corner" (a coordinate position consisting of all the maximal ordinates for each dimension for all points within the envelope).</documentation>
  </annotation>
  <choice>
    <sequence>
      <element name="lowerCorner" type="gml:DirectPositionType"/>
      <element name="upperCorner" type="gml:DirectPositionType"/>
    </sequence>
    <element ref="gml:coord" minOccurs="2" maxOccurs="2">
      <annotation>
        <appinfo>deprecated</appinfo>
        <documentation>deprecated with GML version 3.0</documentation>
      </annotation>
    </element>
    <element ref="gml:pos" minOccurs="2" maxOccurs="2">
      <annotation>
        <appinfo>deprecated</appinfo>
        <documentation>Deprecated with GML version 3.1. Use the explicit properties "lowerCorner" and "upperCorner" instead.</documentation>
      </annotation>
    </element>
    <element ref="gml:coordinates">
      <annotation>

```

```

        <documentation>Deprecated with GML version 3.1.0. Use the explicit properties "lowerCorner" and
"upperCorner" instead.</documentation>
    </annotation>
</element>
</choice>
<attributeGroup ref="gml:SRSReferenceGroup"/>
</complexType>
<!-- ===== -->
<!-- ===== -->
<!-- ===== -->
<!--      The following types and elements are deprecated and should not be used !      -->
<element name="coord" type="gml:CoordType">
    <annotation>
        <documentation>Deprecated with GML 3.0 and included for backwards compatibility with GML 2. Use the "pos"
element instead.</documentation>
    </annotation>
</element>
<complexType name="CoordType">
    <annotation>
        <documentation>Represents a coordinate tuple in one, two, or three dimensions. Deprecated with GML 3.0 and
replaced by DirectPositionType.</documentation>
    </annotation>
    <sequence>
        <element name="X" type="decimal"/>
        <element name="Y" type="decimal" minOccurs="0"/>
        <element name="Z" type="decimal" minOccurs="0"/>
    </sequence>
</complexType>
<!-- ===== -->
<element name="lineStringProperty" type="gml:LineStringPropertyType">
    <annotation>
        <documentation>Deprecated with GML 3.0 and included only for backwards compatibility with GML 2.0. Use
"curveProperty" instead. This property element either references a line string via the XLink-attributes or contains the line string
element.</documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="LineStringPropertyType">
    <annotation>
        <documentation>This type is deprecated with GML 3 and shall not be used. It is included for backwards compatibility
with GML 2. Use CurvePropertyType instead. A property that has a line string as its value domain can either be an appropriate
geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote
includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be
given, but neither both nor none.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:LineString" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup">
        <annotation>
            <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to
reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by
including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the
World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links
between resources; such links can be used to reference remote properties. A simple link element can be used to implement
pointer functionality, and this functionality has been built into various GML 3 elements by including the
gml:AssociationAttributeGroup.</documentation>
        </annotation>
        </attributeGroup>
    </complexType>
<!-- ===== -->
</schema>

```

## C.15 geometryBasic2d.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 2 U (http://www.xmlspy.com) by Clemens Portele (interactive instruments) -->

```

```

<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:sch="http://www.ascc.net/xml/schematron" xmlns:gml="http://www.opengis.net/gml"
xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDefault="qualified" version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:geometryBasic2d:v3.1.0">geometryBasic2d.xsd</appinfo>
    <documentation>
      Copyright (c) 2001-2003 OGC, All Rights Reserved.
    </documentation>
  </annotation>
  <include schemaLocation="geometryBasic0d1d.xsd"/>
  <!-- ===== -->
  <!-- primitive geometry objects (2-dimensional) -->
  <!-- ===== -->
  <element name="_Surface" type="gml:AbstractSurfaceType" abstract="true" substitutionGroup="gml:_GeometricPrimitive">
    <annotation>
      <documentation>The "_Surface" element is the abstract head of the substitution group for all (continuous) surface elements.</documentation>
    </annotation>
    <element name="AbstractSurfaceType">
      <annotation>
        <documentation>
          An abstraction of a surface to support the different levels of complexity. A surface is always a continuous region of a plane.
        </documentation>
      </annotation>
      <complexContent>
        <extension base="gml:AbstractGeometricPrimitiveType"/>
      </complexContent>
    </element>
    <!-- ===== -->
    <element name="surfaceProperty" type="gml:Surface.PropertyType">
      <annotation>
        <appinfo>
          <sch:pattern>
            <sch:rule context="gml:surfaceProperty">
              <sch:extends rule="hrefOrContent"/>
            </sch:rule>
          </sch:pattern>
        </appinfo>
        <documentation>This property element either references a surface via the XLink-attributes or contains the surface element. surfaceProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for _Surface.</documentation>
      </annotation>
    </element>
    <!-- ===== -->
    <element name="Surface.PropertyType">
      <annotation>
        <documentation>A property that has a surface as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>
      </annotation>
      <sequence>
        <element ref="gml:_Surface" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup">
        <annotation>
          <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties.

```

```

<element name="surfaceArrayProperty" type="gml:SurfaceArrayPropertyType"/>
<!-- ===== -->
<complexType name="SurfaceArrayPropertyType">
  <annotation>
    <documentation>A container for an array of surfaces. The elements are always contained in the array property, referencing geometry elements or arrays of geometry elements is not supported.</documentation>
  </annotation>
  <sequence>
    <element ref="gml:_Surface" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<!-- ===== -->
<element name="Polygon" type="gml:PolygonType" substitutionGroup="gml:_Surface"/>
<!-- ===== -->
<complexType name="PolygonType">
  <annotation>
    <documentation>A Polygon is a special surface that is defined by a single surface patch. The boundary of this patch is coplanar and the polygon uses planar interpolation in its interior. It is backwards compatible with the Polygon of GML 2, GM_Polygon of ISO 19107 is implemented by PolygonPatch.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractSurfaceType">
      <sequence>
        <element ref="gml:exterior" minOccurs="0"/>
        <element ref="gml:interior" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- rings (closed curves for surface boundaries) -->
<!-- ===== -->
<element name="_Ring" type="gml:AbstractRingType" abstract="true" substitutionGroup="gml:_Geometry">
  <annotation>
    <documentation>The "_Ring" element is the abstract head of the substitution group for all closed boundaries of a surface patch.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractRingType" abstract="true">
  <annotation>
    <documentation>An abstraction of a ring to support surface boundaries of different complexity.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGeometryType"/>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="exterior" type="gml:AbstractRingPropertyType">
  <annotation>
    <documentation>A boundary of a surface consists of a number of rings. In the normal 2D case, one of these rings is distinguished as being the exterior boundary. In a general manifold this is not always possible, in which case all boundaries shall be listed as interior boundaries, and the exterior will be empty.</documentation>
  </annotation>
</element>
<element name="interior" type="gml:AbstractRingPropertyType">
  <annotation>
    <documentation>A boundary of a surface consists of a number of rings. The "interior" rings separate the surface / surface patch from the area enclosed by the rings.</documentation>
  </annotation>
</element>
<element name="outerBoundaryIs" type="gml:AbstractRingPropertyType" substitutionGroup="gml:exterior">
  <annotation>
    <documentation>Deprecated with GML 3.0, included only for backwards compatibility with GML 2. Use "exterior" instead.</documentation>
  </annotation>
</element>

```

```

</element>
<element name="innerBoundaryls" type="gml:AbstractRingPropertyType" substitutionGroup="gml:interior">
  <annotation>
    <documentation>
      Deprecated with GML 3.0, included only for backwards compatibility with GML 2. Use "interior" instead.
    </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractRingPropertyType">
  <annotation>
    <documentation>
      Encapsulates a ring to represent the surface boundary property of a surface.
    </documentation>
  </annotation>
  <sequence>
    <element ref="gml:_Ring"/>
  </sequence>
</complexType>
<!-- ===== -->
<element name="LinearRing" type="gml:LinearRingType" substitutionGroup="gml:_Ring"/>
<!-- ===== -->
<complexType name="LinearRingType">
  <annotation>
    <documentation>A LinearRing is defined by four or more coordinate tuples, with linear interpolation between them; the first and last coordinates must be coincident.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractRingType">
      <sequence>
        <choice>
          <annotation>
            <documentation>GML supports two different ways to specify the control points of a linear ring.
1. A sequence of "pos" (DirectPositionType) or "pointProperty" (PointPropertyType) elements. "pos" elements are control points that are only part of this ring, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this ring (reuse of existing points).
2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this ring only. The number of direct positions in the list must be at least four.</documentation>
            </annotation>
          <choice minOccurs="4" maxOccurs="unbounded">
            <element ref="gml:pos"/>
            <element ref="gml:pointProperty"/>
            <element ref="gml:pointRep">
              <annotation>
                <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included for backwards compatibility with GML 3.0.0.</documentation>
              </annotation>
            </element>
          </choice>
          <element ref="gml:posList"/>
          <element ref="gml:coordinates">
            <annotation>
              <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
            </annotation>
          </element>
          <element ref="gml:coord" minOccurs="4" maxOccurs="unbounded">
            <annotation>
              <documentation>Deprecated with GML version 3.0 and included for backwards compatibility with GML 2. Use "pos" elements instead.</documentation>
            </annotation>
          </element>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<complexType name="LinearRingPropertyType">
  <annotation>
    <documentation>

```

```

        Encapsulates a ring to represent properties in features or geometry collections.
    </documentation>
</annotation>
<choice>
    <element ref="gml:LinearRing"/>
</choice>
</complexType>
<!-- ===== -->
<!--

```

The following types and elements are deprecated and should not be used !

```

-->
<!-- ===== -->
<element name="polygonProperty" type="gml:PolygonPropertyType">
    <annotation>
        <documentation>Deprecated with GML 3.0 and included only for backwards compatibility with GML 2.0. Use "surfaceProperty" instead.
This property element either references a polygon via the XLink-attributes or contains the polygon element.</documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="PolygonPropertyType">
    <annotation>
        <documentation>This type is deprecated with GML 3 and shall not be used. It is included for backwards compatibility with GML 2. Use SurfacePropertyType instead.
A property that has a polygon as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:Polygon" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup">
        <annotation>
            <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties.
A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the gml:AssociationAttributeGroup.</documentation>
        </annotation>
        </attributeGroup>
    </complexType>
    <!-- ===== -->
</schema>
```

## C.16 geometryComplexes.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:sch="http://www.ascc.net/xml/schematron" xmlns:gml="http://www.opengis.net/gml"
xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDefault="qualified" version="3.1.0">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-
xsd:geometryComplexes:v3.1.0">geometryComplexes.xsd</appinfo>
        <documentation>
            Copyright (c) 2001-2002 OGC, All Rights Reserved.
        </documentation>
        </annotation>
        <include schemaLocation="geometryAggregates.xsd"/>
    <!-- ===== -->
    <element name="CompositeCurve" type="gml:CompositeCurveType" substitutionGroup="gml:_Curve"/>
    <!-- ===== -->
    <complexType name="CompositeCurveType">
        <annotation>
            <documentation>
```

A CompositeCurve is defined by a sequence of (orientable) curves such that the each curve in the sequence terminates at the start point of the subsequent curve in the list.

```

</documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractCurveType">
    <sequence>
      <element ref="gml:curveMember" maxOccurs="unbounded">
        <annotation>
          <documentation>This element references or contains one curve in the composite curve. The curves are contiguous, the collection of curves is ordered.
        </documentation>
        NOTE: This definition allows for a nested structure, i.e. a CompositeCurve may use, for example, another CompositeCurve as a curve member.</documentation>
        <annotation>
          </annotation>
        <element>
          </element>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
<!-- ===== -->
<complexType name="CompositeCurvePropertyType">
  <sequence>
    <element ref="gml:CompositeCurve" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<element name="CompositeSurface" type="gml:CompositeSurfaceType" substitutionGroup="gml:_Surface"/>
<!-- ===== -->
<complexType name="CompositeSurfaceType">
  <annotation>
    <documentation>A CompositeSurface is defined by a set of orientable surfaces. A composite surface is geometry type with all the geometric properties of a (primitive) surface. Essentially, a composite surface is a collection of surfaces that join in pairs on common boundary curves and which, when considered as a whole, form a single surface. </documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractSurfaceType">
        <sequence>
          <element ref="gml:surfaceMember" maxOccurs="unbounded">
            <annotation>
              <documentation>This element references or contains one surface in the composite surface. The surfaces are contiguous.
            </documentation>
            NOTE: This definition allows for a nested structure, i.e. a CompositeSurface may use, for example, another CompositeSurface as a member.</documentation>
            <annotation>
              </annotation>
            <element>
              </element>
            </sequence>
          </extension>
        </complexContent>
      </complexType>
<!-- ===== -->
<complexType name="CompositeSurfacePropertyType">
  <sequence>
    <element ref="gml:CompositeSurface" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<element name="CompositeSolid" type="gml:CompositeSolidType" substitutionGroup="gml:_Solid"/>
<!-- ===== -->
<complexType name="CompositeSolidType">
  <annotation>
    <documentation>
      A composite solid is a geometry type with all the geometric properties of a (primitive) solid.
      Essentially, a composite solid is a collection of solids that join in pairs on common boundary surfaces and which, when considered as a whole, form a single solid.
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractSolidType">
      <sequence>

```

```

<element ref="gml:solidMember" maxOccurs="unbounded">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:solidMember">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
  </annotation>

```

<documentation>This element references or contains one solid in the composite solid. The solids are contiguous.

NOTE: This definition allows for a nested structure, i.e. a CompositeSolid may use, for example, another CompositeSolid as a member.</documentation>

```

    </annotation>
    </element>
  </sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== -->

```

```

<complexType name="CompositeSolidPropertyType">
  <sequence>
    <element ref="gml:CompositeSolid" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- complex/composite geometry objects -->
<!-- ===== -->

```

```

<element name="GeometricComplex" type="gml:GeometricComplexType" substitutionGroup="gml:_Geometry"/>
<!-- ===== -->
<complexType name="GeometricComplexType">
  <annotation>

```

A geometric complex.  
</documentation>

```

    <complexContent>
      <extension base="gml:AbstractGeometryType">
        <sequence>
          <element name="element" type="gml:GeometricPrimitivePropertyType" maxOccurs="unbounded"/>
        </sequence>
      </extension>
    </complexContent>

```

```

</complexType>
<!-- ===== -->
<complexType name="GeometricComplexPropertyType">
  <annotation>

```

<documentation>A property that has a geometric complex as its value domain can either be an appropriate geometry element encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither both nor none.

NOTE: The allowed geometry elements contained in such a property (or referenced by it) have to be modelled by an XML Schema choice element since the composites inherit both from geometric complex \*and\* geometric primitive and are already part of the \_GeometricPrimitive substitution group.</documentation>

```

    </annotation>
    <choice minOccurs="0">
      <element ref="gml:GeometricComplex"/>
      <element ref="gml:CompositeCurve"/>
      <element ref="gml:CompositeSurface"/>
      <element ref="gml:CompositeSolid"/>
    </choice>
    <attributeGroup ref="gml:AssociationAttributeGroup">
      <annotation>

```

<documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links between resources; such links can be used to reference remote properties.

A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3 elements by including the `gml:AssociationAttributeGroup`.

```

</annotation>
</attributeGroup>
</complexType>
<!-- ===== -->
</schema>
```

## C.17 geometryPrimitives.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 2 U (http://www.xmlspy.com) by Clemens Portele (interactive instruments) -->
<schema targetNamespace="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:gml="http://www.opengis.net/gml" xmlns:sch="http://www.ascc.net/xml/schematron"
  xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-
xsd:geometryPrimitives:v3.1.0">geometryPrimitives.xsd</appinfo>
    <documentation>
      Copyright (c) 2001-2003 OGC, All Rights Reserved.
    </documentation>
    </annotation>
    <!-- ===== -->
    <include schemaLocation="geometryBasic2d.xsd"/>
    <!-- ===== -->
    <element name="Curve" type="gml:CurveType" substitutionGroup="gml:_Curve"/>
    <!-- ===== -->
    <complexType name="CurveType">
      <annotation>
        <documentation>
          Curve is a 1-dimensional primitive. Curves are continuous, connected, and have a measurable length in terms of
          the coordinate system.
          A curve is composed of one or more curve segments. Each curve segment within a curve may be defined using a
          different interpolation method. The curve segments are connected to one another, with the end point of each segment except the
          last being the start point of the next segment in the segment list.
          The orientation of the curve is positive.
        </documentation>
      </annotation>
      <complexContent>
        <extension base="gml:AbstractCurveType">
          <sequence>
            <element ref="gml:segments">
              <annotation>
                <documentation>This element encapsulates the segments of the curve.</documentation>
              </annotation>
            </element>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
    <!-- ===== -->
    <element name="baseCurve" type="gml:Curve.PropertyType">
      <annotation>
        <appinfo>
          <sch:pattern>
            <sch:rule context="gml:baseCurve">
              <sch:extends rule="hrefOrContent"/>
            </sch:rule>
          </sch:pattern>
        </appinfo>
        <documentation>This property element either references a curve via the XLink-attributes or contains the curve
        element. A curve element is any element which is substitutable for "_Curve".</documentation>
      </annotation>
    </element>
    <!-- ===== -->
    <element name="OrientableCurve" type="gml:OrientableCurveType" substitutionGroup="gml:_Curve"/>
    <!-- ===== -->
    <complexType name="OrientableCurveType">
      <annotation>
        <documentation>
```

OrientableCurve consists of a curve and an orientation. If the orientation is "+", then the OrientableCurve is identical to the baseCurve. If the orientation is "-", then the OrientableCurve is related to another \_Curve with a parameterization that reverses the sense of the curve traversal.

```

</documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractCurveType">
    <sequence>
      <element ref="gml:baseCurve">
        <annotation>
          <documentation>References or contains the base curve (positive orientation).</documentation>
        </annotation>
      </element>
    </sequence>
    <attribute name="orientation" type="gml:SignType" default="+">
      <annotation>
        <documentation>If the orientation is "+", then the OrientableCurve is identical to the baseCurve. If the orientation is "-", then the OrientableCurve is related to another _Curve with a parameterization that reverses the sense of the curve traversal. "+" is the default value.</documentation>
      </annotation>
    </attribute>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
<!-- curve segments (1-dimensional) -->
<!-- ===== -->
<!-- ===== -->
<element name="_CurveSegment" type="gml:AbstractCurveSegmentType" abstract="true">
  <annotation>
    <documentation>The "_CurveSegment" element is the abstract head of the substitution group for all curve segment elements, i.e. continuous segments of the same interpolation mechanism.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractCurveSegmentType" abstract="true">
  <annotation>
    <documentation>
      Curve segment defines a homogeneous segment of a curve.
    </documentation>
  </annotation>
  <attribute name="numDerivativesAtStart" type="integer" use="optional" default="0">
    <annotation>
      <documentation>The attribute "numDerivativesAtStart" specifies the type of continuity between this curve segment and its predecessor. If this is the first curve segment in the curve, one of these values, as appropriate, is ignored. The default value of "0" means simple continuity, which is a mandatory minimum level of continuity. This level is referred to as "C 0" in mathematical texts. A value of 1 means that the function and its first derivative are continuous at the appropriate end point: "C 1" continuity. A value of "n" for any integer means the function and its first n derivatives are continuous: "C n" continuity. NOTE: Use of these values is only appropriate when the basic curve definition is an underdetermined system. For example, line string segments cannot support continuity above C 0, since there is no spare control parameter to adjust the incoming angle at the end points of the segment. Spline functions on the other hand often have extra degrees of freedom on end segments that allow them to adjust the values of the derivatives to support C 1 or higher continuity.</documentation>
    </annotation>
  </attribute>
  <attribute name="numDerivativesAtEnd" type="integer" use="optional" default="0">
    <annotation>
      <documentation>The attribute "numDerivativesAtEnd" specifies the type of continuity between this curve segment and its successor. If this is the last curve segment in the curve, one of these values, as appropriate, is ignored. The default value of "0" means simple continuity, which is a mandatory minimum level of continuity. This level is referred to as "C 0" in mathematical texts. A value of 1 means that the function and its first derivative are continuous at the appropriate end point: "C 1" continuity. A value of "n" for any integer means the function and its first n derivatives are continuous: "C n" continuity. NOTE: Use of these values is only appropriate when the basic curve definition is an underdetermined system. For example, line string segments cannot support continuity above C 0, since there is no spare control parameter to adjust the incoming angle at the end points of the segment. Spline functions on the other hand often have extra degrees of freedom on end segments that allow them to adjust the values of the derivatives to support C 1 or higher continuity.</documentation>
    </annotation>
  </attribute>

```

```

<attribute name="numDerivativeInterior" type="integer" use="optional" default="0">
    <annotation>
        <documentation>The attribute "numDerivativesInterior" specifies the type of continuity that is guaranteed interior to the curve. The default value of "0" means simple continuity, which is a mandatory minimum level of continuity. This level is referred to as "C 0" in mathematical texts. A value of 1 means that the function and its first derivative are continuous at the appropriate end point: "C 1" continuity. A value of "n" for any integer means the function and its first n derivatives are continuous: "C n" continuity.


NOTE: Use of these values is only appropriate when the basic curve definition is an underdetermined system. For example, line string segments cannot support continuity above C 0, since there is no spare control parameter to adjust the incoming angle at the end points of the segment. Spline functions on the other hand often have extra degrees of freedom on end segments that allow them to adjust the values of the derivatives to support C 1 or higher continuity.</documentation>


    </annotation>
</attribute>
</complexType>
<!-- ===== -->
<element name="segments" type="gml:CurveSegmentArrayPropertyType">
    <annotation>
        <documentation>This property element contains a list of curve segments. The order of the elements is significant and shall be preserved when processing the array.</documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="CurveSegmentArrayPropertyType">
    <annotation>
        <documentation>A container for an array of curve segments.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:_CurveSegment" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
</complexType>
<!-- ===== global element in "_CurveSegment" substitution group ===== -->
<element name="LineStringSegment" type="gml:LineStringSegmentType" substitutionGroup="gml:_CurveSegment"/>
<!-- ===== -->
<complexType name="LineStringSegmentType">
    <annotation>
        <documentation>
            A LineStringSegment is a curve segment that is defined by two or more coordinate tuples, with linear interpolation between them.
            Note: LineStringSegment implements GM_LineString of ISO 19107.
        </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractCurveSegmentType">
            <sequence>
                <choice>
                    <annotation>
                        <documentation>GML supports two different ways to specify the control points of a curve segment.


1. A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the list must be at least two.</documentation>


                    </annotation>
                    <choice minOccurs="2" maxOccurs="unbounded">
                        <element ref="gml:pos"/>
                        <element ref="gml:pointProperty"/>
                        <element ref="gml:pointRep">
                            <annotation>
                                <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included for backwards compatibility with GML 3.0.0.</documentation>
                            </annotation>
                        </element>
                    </choice>
                    <element ref="gml:posList"/>
                    <element ref="gml:coordinates">
                        <annotation>
                            <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
                        </annotation>
                    </element>
                </choice>
            </sequence>
        </extension>
    </complexContent>
</complexType>

```

```

        </sequence>
        <attribute name="interpolation" type="gml:CurveInterpolationType" fixed="linear">
            <annotation>
                <documentation>The attribute "interpolation" specifies the curve interpolation mechanism used for this
segment. This mechanism
uses the control points and control parameters to determine the position of this curve segment. For a LineStringSegment the
interpolation is fixed as "linear".</documentation>
            </annotation>
        </attribute>
    </extension>
</complexContent>
</complexType>

<element name="ArcString" type="gml:ArcStringType" substitutionGroup="gml:_CurveSegment"/>

<complexType name="ArcStringType">
    <annotation>
        <documentation>
An ArcString is a curve segment that uses three-point circular arc interpolation.
        </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractCurveSegmentType">
            <sequence>
                <choice>
                    <annotation>
                        <documentation>GML supports two different ways to specify the control points of a curve segment.
1. A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points
that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry
elements or reference another point defined outside of this curve segment (reuse of existing points).
2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the
same coordinate reference systems and belong to this curve segment only. The number of direct positions in the list must be at
least three.</documentation>
                    </annotation>
                    <choice minOccurs="3" maxOccurs="unbounded">
                        <element ref="gml:pos"/>
                        <element ref="gml:pointProperty"/>
                        <element ref="gml:pointRep">
                            <annotation>
                                <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included
for backwards compatibility with GML 3.0.0.</documentation>
                            </annotation>
                        </element>
                    </choice>
                    <element ref="gml:posList"/>
                    <element ref="gml:coordinates">
                        <annotation>
                            <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
                        </annotation>
                    </element>
                </choice>
            </sequence>
            <attribute name="interpolation" type="gml:CurveInterpolationType" fixed="circularArc3Points">
                <annotation>
                    <documentation>The attribute "interpolation" specifies the curve interpolation mechanism used for this
segment. This mechanism
uses the control points and control parameters to determine the position of this curve segment. For an ArcString the interpolation
is fixed as "circularArc3Points".</documentation>
                </annotation>
            </attribute>
            <attribute name="numArc" type="integer" use="optional">
                <annotation>
                    <documentation>The number of arcs in the arc string can be explicitly stated in this attribute. The number
of control points in the arc string must be 2 * numArc + 1.</documentation>
                </annotation>
            </attribute>
        </extension>
    </complexContent>
</complexType>

```

```

<!-- ===== global element in "_CurveSegment" substitution group ===== -->
<element name="Arc" type="gml:ArcType" substitutionGroup="gml:ArcString"/>
<!-- ===== -->
<complexType name="ArcType">
  <annotation>
    <documentation>
      An Arc is an arc string with only one arc unit, i.e. three control points.
    </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:ArcStringType">
      <sequence>
        <choice>
          <annotation>
            <documentation>GML supports two different ways to specify the control points of a curve segment.
1. A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points
that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry
elements or reference another point defined outside of this curve segment (reuse of existing points).
2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the
same coordinate reference systems and belong to this curve segment only. The number of direct positions in the list must be
three.</documentation>
          </annotation>
          <choice minOccurs="3" maxOccurs="3">
            <element ref="gml:pos"/>
            <element ref="gml:pointProperty"/>
            <element ref="gml:pointRep"/>
            <annotation>
              <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included
for backwards compatibility with GML 3.0.0.</documentation>
            </annotation>
          </choice>
          <element ref="gml:posList"/>
          <element ref="gml:coordinates">
            <annotation>
              <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
            </annotation>
          </element>
        </choice>
      </sequence>
      <attribute name="numArc" type="integer" use="optional" fixed="1">
        <annotation>
          <documentation>An arc is an arc string consisting of a single arc, the attribute is fixed to
"1".</documentation>
        </annotation>
      </attribute>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== global element in "_CurveSegment" substitution group ===== -->
<element name="Circle" type="gml:CircleType" substitutionGroup="gml:Arc"/>
<!-- ===== -->
<complexType name="CircleType">
  <annotation>
    <documentation>A Circle is an arc whose ends coincide to form a simple closed loop. The "start" and "end" bearing
are equal and shall be the bearing for the first controlPoint listed. The three control points must be distinct non-co-linear points for
the Circle to be unambiguously defined. The arc is simply extended past the third control point until the first control point is
encountered.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:ArcType"/>
  </complexContent>
</complexType>
<!-- ===== global element in "_CurveSegment" substitution group ===== -->
<element name="ArcStringByBulge" type="gml:ArcStringByBulgeType" substitutionGroup="gml:_CurveSegment"/>
<!-- ===== -->
<complexType name="ArcStringByBulgeType">
  <annotation>
    <documentation>
      This variant of the arc computes the mid points of the arcs instead of storing the coordinates directly. The control
      point sequence consists of the start and end points of each arc plus the bulge.
    </documentation>
  </annotation>

```

```

        </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractCurveSegmentType">
            <sequence>
                <choice>
                    <annotation>
                        <documentation>GML supports two different ways to specify the control points of a curve segment.


1. A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).



2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the list must be at least two.</documentation>


                    </annotation>
                    <choice minOccurs="2" maxOccurs="unbounded">
                        <element ref="gml:pos"/>
                        <element ref="gml:pointProperty"/>
                        <element ref="gml:pointRep">
                            <annotation>
                                <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included for backwards compatibility with GML 3.0.0.</documentation>
                            </annotation>
                        </element>
                    </choice>
                    <element ref="gml:posList"/>
                    <element ref="gml:coordinates">
                        <annotation>
                            <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
                        </annotation>
                    </element>
                </choice>
                <element name="bulge" type="double" maxOccurs="unbounded">
                    <annotation>
                        <documentation>The bulge controls the offset of each arc's midpoint. The "bulge" is the real number multiplier for the normal that determines the offset direction of the midpoint of each arc. The length of the bulge sequence is exactly 1 less than the length of the control point array, since a bulge is needed for each pair of adjacent points in the control point array. The bulge is not given by a distance, since it is simply a multiplier for the normal.


The midpoint of the resulting arc is given by: midPoint = ((startPoint + endPoint)/2.0) + bulge*normal</documentation>


                    </annotation>
                </element>
            </sequence>
            <attribute name="normal" type="gml:VectorType" maxOccurs="unbounded">
                <annotation>
                    <documentation>The attribute "normal" is a vector normal (perpendicular) to the chord of the arc, the line joining the first and last point of the arc. In a 2D coordinate system, there are only two possible directions for the normal, and it is often given as a signed real, indicating its length, with a positive sign indicating a left turn angle from the chord line, and a negative sign indicating a right turn from the chord. In 3D, the normal determines the plane of the arc, along with the start and endPoint of the arc.


The normal is usually a unit vector, but this is not absolutely necessary. If the normal is a zero vector, the geometric object becomes equivalent to the straight line between the two end points. The length of the normal sequence is exactly the same as for the bulge sequence, 1 less than the control point sequence length.</documentation>


                </annotation>
            </attribute>
        </sequence>
        <attribute name="interpolation" type="gml:CurveInterpolationType" fixed="circularArc2PointWithBulge">
            <annotation>
                <documentation>The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For an ArcStringByBulge the interpolation is fixed as "circularArc2PointWithBulge".</documentation>
                </annotation>
            </attribute>
            <attribute name="numArc" type="integer" use="optional">
                <annotation>
                    <documentation>The number of arcs in the arc string can be explicitly stated in this attribute. The number of control points in the arc string must be numArc + 1.</documentation>
                </annotation>
            </attribute>
        </extension>
    
```

```

        </complexContent>
    </complexType>
    <!-- ===== global element in "_CurveSegment" substitution group ===== -->
    <element name="ArcByBulge" type="gml:ArcByBulgeType" substitutionGroup="gml:ArcStringByBulge"/>
    <!-- ===== -->
    <complexType name="ArcByBulgeType">
        <annotation>
            <documentation>
                An ArcByBulge is an arc string with only one arc unit, i.e. two control points and one bulge.
            </documentation>
        </annotation>
        <complexContent>
            <restriction base="gml:ArcStringByBulgeType">
                <sequence>
                    <choice>
                        <annotation>
                            <documentation>GML supports two different ways to specify the control points of a curve segment.
                                1. A sequence of "pos" (DirectPositionType) or "pointProperty" (PointPropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
                                2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the list must be two.</documentation>
                            </annotation>
                            <choice minOccurs="2" maxOccurs="2">
                                <element ref="gml:pos"/>
                                <element ref="gml:pointProperty"/>
                                <element ref="gml:pointRep">
                                    <annotation>
                                        <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included for backwards compatibility with GML 3.0.0.</documentation>
                                    </annotation>
                                </element>
                            </choice>
                            <element ref="gml:posList"/>
                            <element ref="gml:coordinates">
                                <annotation>
                                    <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
                                </annotation>
                            </element>
                        </choice>
                        <element name="bulge" type="double">
                            <annotation>
                                <documentation>The bulge controls the offset of each arc's midpoint. The "bulge" is the real number multiplier for the normal that determines the offset direction of the midpoint of each arc. The length of the bulge sequence is exactly 1 less than the length of the control point array, since a bulge is needed for each pair of adjacent points in the control point array. The bulge is not given by a distance, since it is simply a multiplier for the normal.
                                The midpoint of the resulting arc is given by: midPoint = ((startPoint + endPoint)/2.0) + bulge*normal</documentation>
                            </annotation>
                            <annotation>
                                <documentation>The attribute "normal" is a vector normal (perpendicular) to the chord of the arc, the line joining the first and last point of the arc. In a 2D coordinate system, there are only two possible directions for the normal, and it is often given as a signed real, indicating its length, with a positive sign indicating a left turn angle from the chord line, and a negative sign indicating a right turn from the chord. In 3D, the normal determines the plane of the arc, along with the start and endPoint of the arc.
                                The normal is usually a unit vector, but this is not absolutely necessary. If the normal is a zero vector, the geometric object becomes equivalent to the straight line between the two end points. The length of the normal sequence is exactly the same as for the bulge sequence, 1 less than the control point sequence length.</documentation>
                            </annotation>
                            </elements>
                        </sequence>
                        <attribute name="numArc" type="integer" use="optional" fixed="1">
                            <annotation>
                                <documentation>An arc is an arc string consisting of a single arc, the attribute is fixed to "1".</documentation>
                            </annotation>
                            </attribute>
                        </restriction>
                    </complexContent>
    </complexType>

```

```

</complexType>
<!-- ===== global element in "_CurveSegment" substitution group ===== -->
<element name="ArcByCenterPoint" type="gml:ArcByCenterPointType" substitutionGroup="gml:_CurveSegment"/>
<!-- ===== -->
<complexType name="ArcByCenterPointType">
  <annotation>
    <documentation>
      This variant of the arc requires that the points on the arc have to be computed instead of storing the coordinates directly. The control point is the center point of the arc plus the radius and the bearing at start and end. This representation can be used only in 2D.
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <choice>
          <annotation>
            <documentation>GML supports two different ways to specify the control points of a curve segment.

```

```

        </annotation>
    </attribute>
</extension>
</complexContent>
</complexType>
<!-- ===== global element in "_CurveSegment" substitution group =====-->
<element name="CircleByCenterPoint" type="gml:CircleByCenterPointType" substitutionGroup="gml:ArcByCenterPoint"/>
<!-- =====-->
<complexType name="CircleByCenterPointType">
    <annotation>
        <documentation>A CircleByCenterPoint is an ArcByCenterPoint with identical start and end angle to form a full circle.

```

direction of the offset curve from the basis curve. It can be omitted in the 2D case, where the distance can be positive or negative. In that case, distance defines left side (positive distance) or right side (negative distance) with respect to the tangent to the basis curve.

In 3D the basis curve shall have a well defined tangent direction for every point. The offset curve at any point in 3D, the basis curve shall have a well-defined tangent direction for every point. The offset curve at any point (parameter) on the basis curve c is in the direction

s = v x t where v = c.refDirection()

and

t = c.tangent()

For the offset direction to be well-defined, v shall not on any point of the curve be in the same, or opposite, direction as

t.

The default value of the refDirection shall be the local co-ordinate axis vector for elevation, which indicates up for the curve in a geographic sense.

NOTE! If the refDirection is the positive tangent to the local elevation axis ("points upward"), then the offset vector points to the left of the curve when viewed from above.

```
</documentation>
    </annotation>
    </element>
    </sequence>
    </extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="AffinePlacement" type="gml:AffinePlacementType"/>
<!-- ===== -->
<complexType name="AffinePlacementType">
    <annotation>
        <documentation> A placement takes a standard geometric construction and places it in geographic space. It defines a transformation from a constructive parameter space to the co-ordinate space of the co-ordinate reference system being used. Parameter spaces in formulae in this International Standard are given as (u, v) in 2D and (u, v, w) in 3D. Co-ordinate reference systems positions are given in formulae, in this International Standard, by either (x, y) in 2D, or (x, y, z) in 3D.
```

Affine placements are defined by linear transformations from parameter space to the target co-ordinate space. 2-dimensional Cartesian parameter space, (u, v) transforms into 3-dimensional co-ordinate reference systems, (x, y, z) by using an affine transformation, (u, v)  $\rightarrow$  (x, y, z) which is defined :

$$\begin{matrix} x & ux & vx & x_0 \\ & u & & \\ y & = uy & vy & + y_0 \\ & v & & \\ x & uz & vz & z_0 \end{matrix}$$

Then, given this equation, the location element of the AffinePlacement is the direct position  $(x_0, y_0, z_0)$ , which is the target position of the origin in  $(u, v)$ . The two reference directions  $(ux, uy, uz)$  and  $(vx, vy, vz)$  are the target directions of the unit vectors at the origin in  $(u, v)$ .

```
</documentation>
    </annotation>
    <sequence>
        <element name="location" type="gml:DirectPositionType">
            <annotation>
                <documentation> The "location" property gives the target of the parameter space origin. This is the vector  $(x_0, y_0, z_0)$  in the formulae above.

</documentation>
            </annotation>
        </element>
        <element name="refDirection" type="gml:VectorType" maxOccurs="unbounded">
            <annotation>
                <documentation> The attribute "refDirection" gives the target directions for the co-ordinate basis vectors of the parameter space. These are the columns of the matrix in the formulae given above. The number of directions given shall be inDimension. The dimension of the directions shall be outDimension.



</documentation>
            </annotation>


```

```

</element>
<element name="inDimension" type="positiveInteger">
  <annotation>
    <documentation> Dimension of the constructive parameter
space.
  </documentation>
  </annotation>
</element>
<element name="outDimension" type="positiveInteger">
  <annotation>
    <documentation> Dimension of the co-ordinate space.
  </documentation>
  </annotation>
</element>
</sequence>
</complexType>
<!-- = global element in "_CurveSegment" substitution group ===== -->
<element name="Clothoid" type="gml:ClothoidType" substitutionGroup="gml:_CurveSegment"/>
<!-- ===== -->
<complexType name="ClothoidType">
  <annotation>
    <documentation> A clothoid, or Cornu's spiral, is plane
curve whose curvature is a fixed function of its length.
In suitably chosen co-ordinates it is given by Fresnel's
integrals.

 $x(t) = \int_0^t \cos(AT^2/2) dT$ 
 $y(t) = \int_0^t \sin(AT^2/2) dT$ 

This geometry is mainly used as a transition curve between
curves of type straight line to circular arc or circular arc
to circular arc. With this curve type it is possible to
achieve a C2-continuous transition between the above mentioned
curve types. One formula for the Clothoid is  $A^2 = R^2 t$  where
A is constant, R is the varying radius of curvature along the
the curve and t is the length along and given in the Fresnel
integrals.
</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <element name="refLocation">
          <complexType>
            <sequence>
              <element ref="gml:AffinePlacement">
                <annotation>
                  <documentation> The "refLocation" is an affine mapping
that places the curve defined by the Fresnel Integrals
into the co-ordinate reference system of this object.
                </documentation>
              </annotation>
            </sequence>
          </complexType>
        </element>
        <element name="scaleFactor" type="decimal">
          <annotation>
            <documentation>The element gives the value for the
constant in the Fresnel's integrals.
          </documentation>
        </annotation>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<element name="startParameter" type="double">
  <annotation>
    <documentation>The startParameter is the arc length
distance from the inflection point that will be the start
point for this curve segment. This shall be lower limit
used in the Fresnel integral and is the value of the
constructive parameter of this curve segment at its start

```

point. The startParameter can either be positive or negative.

NOTE! If 0.0 (zero), lies between the startParameter and the endParameter of the clothoid, then the curve goes through the clothoid's inflection point, and the direction of its radius of curvature, given by the second derivative vector, changes sides with respect to the tangent vector. The term length distance for the

```

</documentation>
    </annotation>
</element>
<element name="endParameter" type="double">
    <annotation>
        <documentation>The endParameter is the arc length
distance from the inflection point that will be the end
point for this curve segment. This shall be upper limit
used in the Fresnel integral and is the value of the
constructive parameter of this curve segment at its
start point. The startParameter can either be positive
or negative.
</documentation>
    </annotation>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
<!-- = global element in "_CurveSegment" substitution group = --&gt;
&lt;element name="GeodesicString" type="gml:GeodesicStringType" substitutionGroup="gml:_CurveSegment"/&gt;
<!-- ===== --&gt;
&lt;complexType name="GeodesicStringType"&gt;
    &lt;annotation&gt;
        &lt;documentation&gt; A GeodesicString consists of sequence of
geodesic segments. The type essentially combines a sequence of
Geodesic into a single object.
The GeodesicString is computed from two or more positions and an
interpolation using geodesics defined from the geoid (or
ellipsoid) of the co-ordinate reference system being used.
&lt;/documentation&gt;
    &lt;/annotation&gt;
    &lt;complexContent&gt;
        &lt;extension base="gml:AbstractCurveSegmentType"&gt;
            &lt;choice&gt;
                &lt;element ref="gml:posList"/&gt;
                &lt;group ref="gml:geometricPositionGroup" minOccurs="2" maxOccurs="unbounded"/&gt;
            &lt;/choice&gt;
            &lt;attribute name="interpolation" type="gml:CurveInterpolationType" fixed="geodesic"&gt;
                &lt;annotation&gt;
                    &lt;documentation&gt;The attribute "interpolation" specifies the
curve interpolation mechanism used for this segment. This
mechanism uses the control points and control parameters to
determine the position of this curve segment. For an
GeodesicString the interpolation is fixed as "geodesic".
&lt;/documentation&gt;
                &lt;/annotation&gt;
            &lt;/attribute&gt;
        &lt;/extension&gt;
    &lt;/complexContent&gt;
&lt;/complexType&gt;
<!-- = global element in "_CurveSegment" substitution group = --&gt;
&lt;element name="Geodesic" type="gml:GeodesicType" substitutionGroup="gml:GeodesicString"/&gt;
<!-- ===== --&gt;
&lt;complexType name="GeodesicType"&gt;
    &lt;annotation&gt;
        &lt;documentation&gt; A Geodesic consists of two distinct
positions joined by a geodesic curve. The control points of
a Geodesic shall lie on the geodesic between its start
point and end points. Between these two points, a geodesic
curve defined from ellipsoid or geoid model used by the
</pre>

```

co-ordinate reference systems may be used to interpolate other positions. Any other point in the controlPoint array must fall on this geodesic.

```
</documentation>
</annotation>
<complexContent>
  <extension base="gml:GeodesicStringType"/>
</complexContent>
</complexType>
<!-- ===== global element in "_CurveSegment" substitution group ===== -->
<element name="CubicSpline" type="gml:CubicSplineType" substitutionGroup="gml:_CurveSegment"/>
<!-- ===== -->
<complexType name="CubicSplineType">
  <annotation>
    <documentation>
```

Cubic splines are similar to line strings in that they are a sequence of segments each with its own defining function. A cubic spline uses the control points and a set of derivative parameters to define a piecewise 3rd degree polynomial interpolation. Unlike line-strings, the parameterization by arc length is not necessarily still a polynomial.

The function describing the curve must be C2, that is, have a continuous 1st and 2nd derivative at all points, and pass through the controlPoints in the order given. Between the control points, the curve segment is defined by a cubic polynomial. At each control point, the polynomial changes in such a manner that the 1st and 2nd derivative vectors are the same from either side. The control parameters record must contain vectorAtStart, and vectorAtEnd which are the unit tangent vectors at controlPoint[1] and controlPoint[n] where n = controlPoint.count.

Note: only the direction of the vectors is relevant, not their length.

```
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <choice>
          <annotation>
            <documentation>GML supports two different ways to specify the control points of a curve segment.
```

1. A sequence of "pos" (DirectPositionType) or "pointProperty" (PointPropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).
2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only. The number of direct positions in the list must be at least three.

```
</choice>
<annotation>
  <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included for backwards compatibility with GML 3.0.0.</documentation>
</annotation>
</element>
</choice>
<element ref="gml:posList"/>
<element ref="gml:coordinates">
  <annotation>
    <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
  </annotation>
</element>
</choice>
<element name="vectorAtStart" type="gml:VectorType">
  <annotation>
    <documentation>"vectorAtStart" is the unit tangent vector at the start point of the spline.</documentation>
  </annotation>
</element>
<element name="vectorAtEnd" type="gml:VectorType">
  <annotation>
    <documentation>"vectorAtEnd" is the unit tangent vector at the end point of the spline.</documentation>
  </annotation>
</element>
</sequence>
<attribute name="interpolation" type="gml:CurveInterpolationType" fixed="cubicSpline">
  <annotation>
```

`<documentation>The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For a CubicSpline the interpolation is fixed as "cubicSpline".</documentation>`

```

</annotation>
</attribute>
<attribute name="degree" type="integer" fixed="3">
    <annotation>
        <documentation>The degree for a cubic spline is "3".</documentation>
    </annotation>
</attribute>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<complexType name="KnotType">
    <annotation>
        <documentation>A knot is a breakpoint on a piecewise spline curve.</documentation>
    </annotation>
    <sequence>
        <element name="value" type="double">
            <annotation>
                <documentation>The property "value" is the value of the parameter at the knot of the spline. The sequence of knots shall be a non-decreasing sequence. That is, each knot's value in the sequence shall be equal to or greater than the previous knot's value. The use of equal consecutive knots is normally handled using the multiplicity.</documentation>
            </annotation>
        </element>
        <element name="multiplicity" type="nonNegativeInteger">
            <annotation>
                <documentation>The property "multiplicity" is the multiplicity of this knot used in the definition of the spline (with the same weight).</documentation>
            </annotation>
        </element>
        <element name="weight" type="double">
            <annotation>
                <documentation>The property "weight" is the value of the averaging weight used for this knot of the spline.</documentation>
            </annotation>
        </element>
    </sequence>
</complexType>
<!-- ===== -->
<complexType name="Knot.PropertyType">
    <annotation>
        <documentation>Encapsulates a knot to use it in a geometric type.</documentation>
    </annotation>
    <sequence>
        <element name="Knot" type="gml:KnotType"/>
    </sequence>
</complexType>
<!-- ===== global element in "_CurveSegment" substitution group ===== -->
<element name="BSpline" type="gml:BSplineType" substitutionGroup="gml:_CurveSegment"/>
<!-- ===== -->
<complexType name="BSplineType">
    <annotation>
        <documentation>A B-Spline is a piecewise parametric polynomial or rational curve described in terms of control points and basis functions. Knots are breakpoints on the curve that connect its pieces. They are given as a non-decreasing sequence of real numbers. If the weights in the knots are equal then it is a polynomial spline. The degree is the algebraic degree of the basis functions. </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractCurveSegmentType">
            <sequence>
                <choice>
                    <annotation>
                        <documentation>GML supports two different ways to specify the control points of a curve segment.</documentation>
                    </annotation>

```

1. A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this curve segment (reuse of existing points).

2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this curve segment only.

```

</annotation>
<choice minOccurs="0" maxOccurs="unbounded">
    <element ref="gml:pos"/>
    <element ref="gml:pointProperty"/>
    <element ref="gml:pointRep">
        <annotation>
            <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included for backwards compatibility with GML 3.0.0.</documentation>
        </annotation>
    </element>
</choice>
<element ref="gml:posList"/>
<element ref="gml:coordinates">
    <annotation>
        <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
    </annotation>
</element>
</choice>
<element name="degree" type="nonNegativeInteger">
    <annotation>
        <documentation>The attribute "degree" shall be the degree of the polynomial used for interpolation in this spline.</documentation>
    </annotation>
</element>
<element name="knot" type="gml:Knot.PropertyType" minOccurs="2" maxOccurs="unbounded">
    <annotation>
        <documentation>The property "knot" shall be the sequence of distinct knots used to define the spline basis functions.</documentation>
    </annotation>
</element>
</sequence>
<attribute name="interpolation" type="gml:CurveInterpolationType" default="polynomialSpline">
    <annotation>
        <documentation>The attribute "interpolation" specifies the curve interpolation mechanism used for this segment. This mechanism uses the control points and control parameters to determine the position of this curve segment. For a BSpline the interpolation can be either "polynomialSpline" or "rationalSpline", default is "polynomialSpline".</documentation>
    </annotation>
</attribute>
<attribute name="isPolynomial" type="boolean" use="optional">
    <annotation>
        <documentation>The attribute "isPolynomial" is set to "true" if this is a polynomial spline.</documentation>
    </annotation>
</attribute>
<attribute name="knotType" type="gml:KnotTypesType" use="optional">
    <annotation>
        <documentation>The attribute "knotType" gives the type of knot distribution used in defining this spline.</documentation>
    </annotation>
</attribute>

```

This is for information only  
and is set according to the different construction-functions.

```

</annotation>
</extension>
</complexContent>
</complexType>
<!-- ===== global element in "_CurveSegment" substitution group =====-->
<element name="Bezier" type="gml:BezierType" substitutionGroup="gml:BSpline"/>
<!-- =====-->
<complexType name="BezierType">
    <annotation>
        <documentation>Bezier curves are polynomial splines that use Bezier or Bernstein polynomials for interpolation purposes. It is a special case of the B-Spline curve with two knots.</documentation>
    </annotation>
    <restriction base="gml:BSplineType">
        <sequence>

```

```

<choice>
  <annotation>
    <documentation>GML supports two different ways to specify the control points of a curve segment.
    1. A sequence of "pos" (DirectPositionType) or "pointProperty" (Point.PropertyType) elements. "pos" elements are control points
       that are only part of this curve segment, "pointProperty" elements contain a point that may be referenced from other geometry
       elements or reference another point defined outside of this curve segment (reuse of existing points).
    2. The "posList" element allows for a compact way to specify the coordinates of the control points, if all control points are in the
       same coordinate reference systems and belong to this curve segment only.</documentation>
  </annotation>
<choice minOccurs="0" maxOccurs="unbounded">
  <element ref="gml:pos"/>
  <element ref="gml:pointProperty"/>
  <element ref="gml:pointRep">
    <annotation>
      <documentation>Deprecated with GML version 3.1.0. Use "pointProperty" instead. Included
       for backwards compatibility with GML 3.0.0.</documentation>
    </annotation>
  </element>
</choice>
<element ref="gml:posList"/>
<element ref="gml:coordinates">
  <annotation>
    <documentation>Deprecated with GML version 3.1.0. Use "posList" instead.</documentation>
  </annotation>
</element>
</choice>
<element name="degree" type="nonNegativeInteger">
  <annotation>
    <documentation>The attribute "degree" shall be the degree of the polynomial used for interpolation in
       this spline.</documentation>
  </annotation>
</element>
<element name="knot" type="gml:Knot.PropertyType" minOccurs="2" maxOccurs="2">
  <annotation>
    <documentation>The property "knot" shall be the sequence of distinct knots used to define the spline
       basis functions.</documentation>
  </annotation>
</element>
</sequence>
<attribute name="interpolation" type="gml:CurveInterpolationType" fixed="polynomialSpline">
  <annotation>
    <documentation>The attribute "interpolation" specifies the curve interpolation mechanism used for this
       segment. This mechanism
       uses the control points and control parameters to determine the position of this curve segment. For a Bezier the interpolation is
       fixed as "polynomialSpline".</documentation>
  </annotation>
</attribute>
<attribute name="isPolynomial" type="boolean" fixed="true">
  <annotation>
    <documentation>The attribute "isPolynomial" is set to "true" as this is a polynomial
       spline.</documentation>
  </annotation>
</attribute>
<attribute name="knotType" type="gml:KnotTypesType" use="prohibited">
  <annotation>
    <documentation>The property "knotType" is not relevant for Bezier curve segments.</documentation>
  </annotation>
</attribute>
</restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="Surface" type="gml:SurfaceType" substitutionGroup="gml:_Surface"/>
<!-- ===== -->
<complexType name="SurfaceType">
  <annotation>
    <documentation>
      A Surface is a 2-dimensional primitive and is composed of one or more surface patches. The surface patches are
       connected to one another.
    </documentation>
  </annotation>

```

The orientation of the surface is positive ("up"). The orientation of a surface chooses an "up" direction through the choice of the upward normal, which, if the surface is not a cycle, is the side of the surface from which the exterior boundary appears counterclockwise. Reversal of the surface orientation reverses the curve orientation of each boundary component, and interchanges the conceptual "up" and "down" direction of the surface. If the surface is the boundary of a solid, the "up" direction is usually outward. For closed surfaces, which have no boundary, the up direction is that of the surface patches, which must be consistent with one another. Its included surface patches describe the interior structure of the Surface.

```

    </documentation>
</annotation>
<complexContent>
<extension base="gml:AbstractSurfaceType">
    <sequence>
        <element ref="gml:patches">
            <annotation>
                <documentation>This element encapsulates the patches of the surface.</documentation>
            </annotation>
        </element>
    </sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="baseSurface" type="gml:SurfacePropertyType">
    <annotation>
        <appinfo>
            <sch:pattern>
                <sch:rule context="gml:baseSurface">
                    <sch:extends rule="hrefOrContent"/>
                </sch:rule>
            </sch:pattern>
        </appinfo>
        <documentation>This property element either references a surface via the XLink-attributes or contains the surface element. A surface element is any element which is substitutable for "_Surface".</documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="OrientableSurface" type="gml:OrientableSurfaceType" substitutionGroup="gml:_Surface"/>
<!-- ===== -->
<complexType name="OrientableSurfaceType">
    <annotation>
        <documentation>
            OrientableSurface consists of a surface and an orientation. If the orientation is "+", then the OrientableSurface is identical to the baseSurface. If the orientation is "-", then the OrientableSurface is a reference to a Surface with an up-normal that reverses the direction for this OrientableSurface, the sense of "the top of the surface".
        </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractSurfaceType">
            <sequence>
                <element ref="gml:baseSurface">
                    <annotation>
                        <documentation>References or contains the base surface (positive orientation).</documentation>
                    </annotation>
                </element>
            </sequence>
            <attribute name="orientation" type="gml:SignType" default="+">
                <annotation>
                    <documentation>If the orientation is "+", then the OrientableSurface is identical to the baseSurface. If the orientation is "-", then the OrientableSurface is a reference to a Surface with an up-normal that reverses the direction for this OrientableSurface, the sense of "the top of the surface". "+" is the default value.</documentation>
                </annotation>
            </attribute>
        </extension>
    </complexContent>
</complexType>
<!-- ===== -->
<!-- surface patches (2-dimensional) -->
<!-- ===== -->
<!-- ===== -->
<element name="_SurfacePatch" type="gml:AbstractSurfacePatchType" abstract="true">
    <annotation>

```

```

<documentation>The "_SurfacePatch" element is the abstract head of the substitution group for all surface patch elements describing a continuous portion of a surface.</documentation>
</annotation>
</element>
<!-- ===== -->
<complexType name="AbstractSurfacePatchType" abstract="true">
  <annotation>
    <documentation>
      A surface patch defines a homogenous portion of a surface.
    </documentation>
  </annotation>
</complexType>
<!-- ===== -->
<element name="patches" type="gml:SurfacePatchArrayPropertyType">
  <annotation>
    <documentation>This property element contains a list of surface patches. The order of the elements is significant and shall be preserved when processing the array.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="SurfacePatchArrayPropertyType">
  <annotation>
    <documentation>A container for an array of surface patches.</documentation>
  </annotation>
  <sequence>
    <element ref="gml:_SurfacePatch" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<!-- ===== -->
<element name="PolygonPatch" type="gml:PolygonPatchType" substitutionGroup="gml:_SurfacePatch"/>
<!-- ===== -->
<complexType name="PolygonPatchType">
  <annotation>
    <documentation>A PolygonPatch is a surface patch that is defined by a set of boundary curves and an underlying surface to which these curves adhere. The curves are coplanar and the polygon uses planar interpolation in its interior. Implements GM_Polygon of ISO 19107.
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractSurfacePatchType">
      <sequence>
        <element ref="gml:exterior" minOccurs="0"/>
        <element ref="gml:interior" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="interpolation" type="gml:SurfaceInterpolationType" fixed="planar">
        <annotation>
          <documentation>The attribute "interpolation" specifies the interpolation mechanism used for this surface patch. Currently only planar surface patches are defined in GML 3, the attribute is fixed to "planar", i.e. the interpolation method shall return points on a single plane. The boundary of the patch shall be contained within that plane.</documentation>
        </annotation>
      </attribute>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="Triangle" type="gml:TriangleType" substitutionGroup="gml:_SurfacePatch"/>
<!-- ===== -->
<complexType name="TriangleType">
  <annotation>
    <documentation>Represents a triangle as a surface with an outer boundary consisting of a linear ring. Note that this is a polygon (subtype) with no inner boundaries. The number of points in the linear ring must be four.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractSurfacePatchType">
      <sequence>
        <element ref="gml:exterior">
          <annotation>
            <documentation>
```

Constraint: The Ring shall be a LinearRing and must form a triangle, the first and the last position must be co-incident.

```

        </documentation>
    </annotation>
</element>
</sequence>
<attribute name="interpolation" type="gml:SurfaceInterpolationType" fixed="planar">
    <annotation>
        <documentation>The attribute "interpolation" specifies the interpolation mechanism used for this surface patch. Currently only planar surface patches are defined in GML 3, the attribute is fixed to "planar", i.e. the interpolation method shall return points on a single plane. The boundary of the patch shall be contained within that plane.</documentation>
    </annotation>
</attribute>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<element name="Rectangle" type="gml:RectangleType" substitutionGroup="gml:_SurfacePatch"/>
<!-- ===== -->

```

<complexType name="RectangleType">  
 <annotation>  
 <documentation>Represents a rectangle as a surface with an outer boundary consisting of a linear ring. Note that this is a polygon (subtype) with no inner boundaries. The number of points in the linear ring must be five.</documentation>
 </annotation>
 <complexContent>

```

        <extension base="gml:AbstractSurfacePatchType">
            <sequence>
                <element ref="gml:exterior">
                    <annotation>
                        <documentation>
                            Constraint: The Ring shall be a LinearRing and must form a rectangle; the first and the last

```

position must be co-incident.

```

                            </documentation>
                        </annotation>
                    </element>
                </sequence>
                <attribute name="interpolation" type="gml:SurfaceInterpolationType" fixed="planar">
                    <annotation>
                        <documentation>The attribute "interpolation" specifies the interpolation mechanism used for this surface patch. Currently only planar surface patches are defined in GML 3, the attribute is fixed to "planar", i.e. the interpolation method shall return points on a single plane. The boundary of the patch shall be contained within that plane.</documentation>
                    </annotation>
</attribute>
</extension>
</complexContent>
</complexType>
<!-- ===== -->

```

<element name="curveMember" type="gml:Curve.PropertyType">  
 <annotation>  
 <documentation>This property element either references a curve via the XLink-attributes or contains the curve element. A curve element is any element which is substitutable for "\_Curve".</documentation>
 </annotation>
</element>
<!-- ===== -->
<element name="Ring" type="gml:RingType" substitutionGroup="gml:\_Ring"/>
<!-- ===== -->

<complexType name="RingType">  
 <annotation>  
 <documentation>A Ring is used to represent a single connected component of a surface boundary. It consists of a sequence of curves connected in a cycle (an object whose boundary is empty).

A Ring is structurally similar to a composite curve in that the endPoint of each curve in the sequence is the startPoint of the next curve in the Sequence. Since the sequence is circular, there is no exception to this rule. Each ring, like all boundaries, is a cycle and each ring is simple.

NOTE: Even though each Ring is simple, the boundary need not be simple. The easiest case of this is where one of the interior rings of a surface is tangent to its exterior ring.</documentation>

```

    </annotation>
    <complexContent>
        <extension base="gml:AbstractRingType">
            <sequence>
                <element ref="gml:curveMember" maxOccurs="unbounded">
                    <annotation>
```

<documentation>This element references or contains one curve in the composite curve. The curves are contiguous, the collection of curves is ordered.

NOTE: This definition allows for a nested structure, i.e. a CompositeCurve may use, for example, another CompositeCurve as a curve member.</documentation>

```
    </annotation>
    </element>
    </sequence>
    </extension>
</complexContent>
</complexType>
<!-- ===== -->
<complexType name="RingPropertyType">
    <annotation>
        <documentation>
            Encapsulates a ring to represent properties in features or geometry collections.
        </documentation>
    </annotation>
    <sequence>
        <element ref="gml:Ring"/>
    </sequence>
</complexType>
<!-- ===== -->
<group name="PointGrid">
    <annotation>
        <documentation> Reference points which are organised
        into sequences or grids(sequences of equal length sequences).
    </documentation>
    </annotation>
    <sequence>
        <element name="row" maxOccurs="unbounded">
            <complexType>
                <sequence>
                    <group ref="gml:geometricPositionListGroup"/>
                </sequence>
            </complexType>
        </element>
    </sequence>
</group>
<!-- ===== -->
<element name="_ParametricCurveSurface" type="gml:AbstractParametricCurveSurfaceType" abstract="true"
substitutionGroup="gml:_SurfacePatch"/>
<!-- ===== -->
<complexType name="AbstractParametricCurveSurfaceType">
    <annotation>
        <documentation>
        </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractSurfacePatchType"/>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="_GriddedSurface" type="gml:AbstractGriddedSurfaceType" abstract="true"
substitutionGroup="gml:_ParametricCurveSurface"/>
<!-- ===== -->
<complexType name="AbstractGriddedSurfaceType">
    <annotation>
        <documentation> A gridded surface is a parametric curve
        surface derived from a rectangular grid in the parameter
        space. The rows from this grid are control points for
        horizontal surface curves; the columns are control points
        for vertical surface curves. The working assumption is that
        for a pair of parametric co-ordinates (s, t) that the
        horizontal curves for each integer offset are calculated
        and evaluated at "s". The defines a sequence of control
        points:
    </documentation>

```

cn(s) : s 1 .....columns

From this sequence a vertical curve is calculated for "s", and evaluated at "t". In most cases, the order of calculation (horizontal-vertical vs. vertical-horizontal) does not make a difference. Where it does, the horizontal-vertical order shall be the one used.

Logically, any pair of curve interpolation types can lead to a subtype of GriddedSurface. The following clauses define some most commonly encountered surfaces that can be represented in this manner.

```
</documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractParametricCurveSurfaceType">
    <sequence>
      <group ref="gml:PointGrid">
        <annotation>
          <documentation> This is the double indexed sequence
          of control points, given in row major form.
        </documentation>
        <NOTE!> There is no assumption made about the shape
        of the grid.
        For example, the positions need not effect a "21/2D"
        surface, consecutive points may be equal in any or all
        of the ordinates. Further, the curves in either or both
        directions may close.
      </group>
      <element name="rows" type="integer" minOccurs="0">
        <annotation>
          <documentation> The attribute rows gives the number
          of rows in the parameter grid.
        </documentation>
      </element>
      <element name="columns" type="integer" minOccurs="0">
        <annotation>
          <documentation> The attribute columns gives the number
          of columns in the parameter grid.
        </documentation>
      </element>
    </sequence>
  </extension>
</complexContent>
<!-- ===== -->
<element name="Cone" type="gml:ConeType" substitutionGroup="gml:_GriddedSurface"/>
<!-- ===== -->
<complexType name="ConeType">
  <annotation>
    <documentation> A cone is a gridded surface given as a
    family of conic sections whose control points vary linearly.
    NOTE! A 5-point ellipse with all defining positions identical
    is a point. Thus, a truncated elliptical cone can be given as a
    2x5 set of control points
    ((P1, P1, P1, P1, P1), (P2, P3, P4, P5, P6)). P1 is the apex
    of the cone. P2, P3, P4, P5 and P6 are any five distinct points
    around the base ellipse of the cone. If the horizontal curves
    are circles as opposed to ellipses, the a circular cone can
    be constructed using ((P1, P1, P1), (P2, P3, P4)). The apex must
    not coincide with the other plane.
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGriddedSurfaceType">
      <attribute name="horizontalCurveType" type="gml:CurveInterpolationType" fixed="circularArc3Points"/>
      <attribute name="verticalCurveType" type="gml:CurveInterpolationType" fixed="linear"/>
    </extension>
  </complexContent>
</complexType>
```

```

<!-- ===== -->
<element name="Cylinder" type="gml:CylinderType" substitutionGroup="gml:_GriddedSurface"/>
<!-- ===== -->
<complexType name="CylinderType">
  <annotation>
    <documentation> A cylinder is a gridded surface given as a
family of circles whose positions vary along a set of parallel
lines, keeping the cross sectional horizontal curves of a
constant shape.
    NOTE! Given the same working assumptions as in the previous
note, a Cylinder can be given by two circles, giving us the
control points of the form ((P1, P2, P3),(P4, P5, P6)).
  </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGriddedSurfaceType">
      <attribute name="horizontalCurveType" type="gml:CurveInterpolationType" fixed="circularArc3Points"/>
      <attribute name="verticalCurveType" type="gml:CurveInterpolationType" fixed="linear"/>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="Sphere" type="gml:SphereType" substitutionGroup="gml:_GriddedSurface"/>
<!-- ===== -->
<complexType name="SphereType">
  <annotation>
    <documentation> A sphere is a gridded surface given as a
family of circles whose positions vary linearly along the
axis of the sphere, and whose radius varies in proportion to
the cosine function of the central angle. The horizontal
circles resemble lines of constant latitude, and the vertical
arcs resemble lines of constant longitude.
    NOTE! If the control points are sorted in terms of increasing
longitude, and increasing latitude, the upNormal of a sphere
is the outward normal.
    EXAMPLE If we take a gridded set of latitudes and longitudes
in degrees,(u,v) such as
  </documentation>

```

(-90,-180) (-90,-90) (-90,0) (-90, 90) (-90, 180)  
(-45,-180) (-45,-90) (-45,0) (-45, 90) (-45, 180)  
( 0,-180) ( 0,-90) ( 0,0) ( 0, 90) ( 0, 180)  
( 45,-180) ( 45,-90) ( 45,0) ( 45, 90) ( 45, 180)  
( 90,-180) ( 90,-90) ( 90,0) ( 90, 90) ( 90, 180)

And map these points to 3D using the usual equations (where R is the radius of the required sphere).

$$\begin{aligned}
z &= R \sin u \\
x &= (R \cos u)(\sin v) \\
y &= (R \cos u)(\cos v)
\end{aligned}$$

We have a sphere of Radius R, centred at (0,0), as a gridded surface. Notice that the entire first row and the entire last row of the control points map to a single point in each 3D Euclidean space, North and South poles respectively, and that each horizontal curve closes back on itself forming a geometric cycle. This gives us a metrically bounded (of finite size), topologically unbounded (not having a boundary, a cycle) surface.

```

</documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractGriddedSurfaceType">
    <attribute name="horizontalCurveType" type="gml:CurveInterpolationType" fixed="circularArc3Points"/>
    <attribute name="verticalCurveType" type="gml:CurveInterpolationType" fixed="circularArc3Points"/>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->

```

```

<element name="PolyhedralSurface" type="gml:PolyhedralSurfaceType" substitutionGroup="gml:Surface"/>
<!-- ===== -->
<complexType name="PolyhedralSurfaceType">
  <annotation>
    <documentation> A polyhedral surface is a surface composed
of polygon surfaces connected along their common boundary
curves. This differs from the surface type only in the
restriction on the types of surface patches acceptable.
  </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:SurfaceType">
      <sequence>
        <group ref="gml:StandardObjectProperties"/>
        <element ref="gml:polygonPatches">
          <annotation>
            <documentation> This property encapsulates the patches of
the polyhedral surface.
          </documentation>
        </annotation>
        </element>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="polygonPatches" type="gml:PolygonPatchArrayPropertyType" substitutionGroup="gml:patches">
  <annotation>
    <documentation> This property element contains a list of
polygon patches. The order of the patches is significant and
shall be preserved when processing the list.
  </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="PolygonPatchArrayPropertyType">
  <annotation>
    <documentation> This type defines a container for an array of
polygon patches.
  </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:SurfacePatchArrayPropertyType">
      <sequence>
        <element ref="gml:PolygonPatch" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="trianglePatches" type="gml:TrianglePatchArrayPropertyType" substitutionGroup="gml:patches">
  <annotation>
    <documentation> This property element contains a list of
triangle patches. The order of the patches is significant and
shall be preserved when processing the list.
  </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="TrianglePatchArrayPropertyType">
  <annotation>
    <documentation> This type defines a container for an array of
triangle patches.
  </documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:SurfacePatchArrayPropertyType">
      <sequence>
        <element ref="gml:Triangle" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>

```

```

        </complexContent>
    </complexType>
    <!-- =============== -->
    <element name="TriangulatedSurface" type="gml:TriangulatedSurfaceType" substitutionGroup="gml:Surface"/>
    <!-- =============== -->
    <complexType name="TriangulatedSurfaceType">
        <annotation>
            <documentation> A triangulated surface is a polyhedral
surface that is composed only of triangles. There is no
restriction on how the triangulation is derived.
        </documentation>
        </annotation>
        <complexContent>
            <restriction base="gml:SurfaceType">
                <sequence>
                    <group ref="gml:StandardObjectProperties"/>
                    <element ref="gml:trianglePatches">
                        <annotation>
                            <documentation> This property encapsulates the patches of
the triangulated surface.
                        </documentation>
                    </annotation>
                    <element>
                        </element>
                    </sequence>
                </restriction>
            </complexContent>
        </complexType>
        <!-- =============== -->
        <element name="Tin" type="gml:TinType" substitutionGroup="gml:TriangulatedSurface"/>
        <!-- =============== -->
        <complexType name="TinType">
            <annotation>
                <documentation> A tin is a triangulated surface that uses
the Delaunay algorithm or a similar algorithm complemented with
consideration of breaklines, stoplines, and maximum length of
triangle sides. These networks satisfy the Delaunay's criterion
away from the modifications: For each triangle in the
network, the circle passing through its vertices does not
contain, in its interior, the vertex of any other triangle.
            </documentation>
            </annotation>
            <complexContent>
                <extension base="gml:TriangulatedSurfaceType">
                    <sequence>
                        <element name="stopLines" type="gml:LineStringSegmentArrayPropertyType" minOccurs="0"
maxOccurs="unbounded">
                            <annotation>
                                <documentation> Stoplines are lines where the local
continuity or regularity of the surface is questionable.
In the area of these pathologies, triangles intersecting
a stopline shall be removed from the tin surface, leaving
holes in the surface. If coincidence occurs on surface
boundary triangles, the result shall be a change of the
surface boundary. Stoplines contains all these
pathological segments as a set of line strings.
                            </documentation>
                            </annotation>
                            <element>
                                </element>
                            </sequence>
                            <element name="breakLines" type="gml:LineStringSegmentArrayPropertyType" minOccurs="0"
maxOccurs="unbounded">
                                <annotation>
                                    <documentation> Breaklines are lines of a critical
nature to the shape of the surface, representing local
ridges, or depressions (such as drainage lines) in the
surface. As such their constituent segments must be
included in the tin even if doing so
violates the Delaunay criterion. Break lines contains these
critical segments as a set of line strings.
                                </documentation>
                                </annotation>
                            </element>
                        </sequence>
                    </extension>
                </complexContent>
            </complexType>
        <!-- =============== -->
    
```

```

        </annotation>
    </element>
    <element name="maxLength" type="gml:LengthType">
        <annotation>
            <documentation> Areas of the surface where data is not
sufficiently dense to assure reasonable calculation shall be
removed by adding a retention criterion for triangles based
on the length of their sides. For many triangle sides
exceeding maximum length, the adjacent triangles to that
triangle side shall be removed from the surface.
        </documentation>
        <annotation>
        </annotation>
    </element>
    <element name="controlPoint">
        <annotation>
            <documentation> The corners of the triangles in the TIN
are often referred to as pots. ControlPoint shall contain a
set of the GM_Position used as posts for this TIN. Since each
TIN contains triangles, there must be at least 3 posts. The
order in which these points are given does not affect the
surface that is represented. Application schemas may add
information based on ordering of control points to facilitate
the reconstruction of the TIN from the control points.
        </documentation>
        <annotation>
        </annotation>
        <complexType>
            <choice>
                <element ref="gml:posList"/>
                <group ref="gml:geometricPositionGroup" minOccurs="3" maxOccurs="unbounded"/>
            </choice>
        </complexType>
        <element>
            </sequence>
        </extension>
    </complexType>
</complexType>
<complexType name="LineStringSegmentArrayPropertyType">
    <sequence>
        <element ref="gml:LineStringSegment" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
</complexType>
<!-- ===== -->
<!-- primitive geometry objects (3-dimensional) -->
<!-- ===== -->
<!-- ===== -->
<element name="_Solid" type="gml:AbstractSolidType" abstract="true" substitutionGroup="gml:_GeometricPrimitive">
    <annotation>
        <documentation>The "_Solid" element is the abstract head of the substitution group for all (continuous) solid
elements.</documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractSolidType">
    <annotation>
        <documentation>
            An abstraction of a solid to support the different levels of complexity. A solid is always contiguous.
        </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractGeometricPrimitiveType"/>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="solidProperty" type="gml:Solid.PropertyType">
    <annotation>
        <appinfo>
            <sch:pattern>
                <sch:rule context="gml:solidProperty">
                    <sch:extends rule="hrefOrContent"/>
                </sch:rule>
            </sch:pattern>
        </appinfo>
    </annotation>

```

```

        </appinfo>
        <documentation>This property element either references a solid via the XLink-attributes or contains the solid
element. solidProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has
a property with a value that is substitutable for _Solid.</documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="Solid.PropertyType">
    <annotation>
        <documentation>A property that has a solid as its value domain can either be an appropriate geometry element
encapsulated in an element of this type or an XLink reference to a remote geometry element (where remote includes geometry
elements located elsewhere in the same document). Either the reference or the contained element must be given, but neither
both nor none.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:_Solid" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup">
        <annotation>
            <documentation>This attribute group includes the XLink attributes (see xlink.xsd). XLink is used in GML to
reference remote resources (including those elsewhere in the same document). A simple link element can be constructed by
including a specific set of XLink attributes. The XML Linking Language (XLink) is currently a Proposed Recommendation of the
World Wide Web Consortium. XLink allows elements to be inserted into XML documents so as to create sophisticated links
between resources; such links can be used to reference remote properties.
A simple link element can be used to implement pointer functionality, and this functionality has been built into various GML 3
elements by including the gml:AssociationAttributeGroup.
            </documentation>
        </annotation>
    </attributeGroup>
</complexType>
<!-- ===== -->
<element name="solidArrayProperty" type="gml:SolidArray.PropertyType"/>
<!-- ===== -->
<complexType name="SolidArray.PropertyType">
    <annotation>
        <documentation>A container for an array of solids. The elements are always contained in the array property,
referencing geometry elements or arrays of geometry elements is not supported.</documentation>
    </annotation>
    <sequence>
        <element ref="gml:_Solid" minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
</complexType>
<!-- ===== -->
<element name="Solid" type="gml:SolidType" substitutionGroup="gml:_Solid"/>
<!-- ===== -->
<complexType name="SolidType">
    <annotation>
        <documentation>A solid is the basis for 3-dimensional geometry. The extent of a solid is defined by the boundary
surfaces (shells). A shell is represented by a composite surface, where every shell is used to represent a single connected
component of the boundary of a solid. It consists of a composite surface (a list of orientable surfaces) connected in a topological
cycle (an object whose boundary is empty). Unlike a Ring, a Shell's elements have no natural sort order. Like Rings, Shells are
simple.</documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractSolidType">
            <sequence>
                <element name="exterior" type="gml:Surface.PropertyType" minOccurs="0">
                    <annotation>
                        <appinfo>
                            <sch:pattern>
                                <sch:rule context="gml:exterior">
                                    <sch:extends rule="hrefOrContent"/>
                                </sch:rule>
                            </sch:pattern>
                        </appinfo>
                        <documentation>Boundaries of solids are similar to surface boundaries. In normal 3-dimensional
Euclidean space, one (composite) surface is distinguished as the exterior. In the more general case, this is not always
possible.</documentation>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>

```

```

</element>
<element name="interior" type="gml:SurfacePropertyType" minOccurs="0" maxOccurs="unbounded">
    <annotation>
        <appinfo>
            <sch:pattern>
                <sch:rule context="gml:interior">
                    <sch:extends rule="hrefOrContent"/>
                </sch:rule>
            </sch:pattern>
        </appinfo>
        <documentation>Boundaries of solids are similar to surface boundaries.</documentation>
    </annotation>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<!-- predefined simple types (enumerations, simple typed arrays) -->
<!-- ===== -->
<simpleType name="CurveInterpolationType">
    <annotation>
        <documentation>CurveInterpolationType is a list of codes that may be used to identify the interpolation mechanisms specified by an application schema.</documentation>
    </annotation>
    <restriction base="string">
        <enumeration value="linear"/>
        <enumeration value="geodesic"/>
        <enumeration value="circularArc3Points"/>
        <enumeration value="circularArc2PointWithBulge"/>
        <enumeration value="circularArcCenterPointWithRadius"/>
        <enumeration value="elliptical"/>
        <enumeration value="clothoid"/>
        <enumeration value="conic"/>
        <enumeration value="polynomialSpline"/>
        <enumeration value="cubicSpline"/>
        <enumeration value="rationalSpline"/>
    </restriction>
</simpleType>
<!-- ===== -->
<simpleType name="SurfaceInterpolationType">
    <annotation>
        <documentation>SurfaceInterpolationType is a list of codes that may be used to identify the interpolation mechanisms specified by an application schema.</documentation>
    </annotation>
    <restriction base="string">
        <enumeration value="none"/>
        <enumeration value="planar"/>
        <enumeration value="spherical"/>
        <enumeration value="elliptical"/>
        <enumeration value="conic"/>
        <enumeration value="tin"/>
        <enumeration value="parametricCurve"/>
        <enumeration value="polynomialSpline"/>
        <enumeration value="rationalSpline"/>
        <enumeration value="triangulatedSpline"/>
    </restriction>
</simpleType>
<!-- ===== -->
<simpleType name="KnotTypesType">
    <annotation>
        <documentation>Defines allowed values for the knots` type. Uniform knots implies that all knots are of multiplicity 1 and they differ by a positive constant from the preceding knot. Knots are quasi-uniform iff they are of multiplicity (degree + 1) at the ends, of multiplicity 1 elsewhere, and they differ by a positive constant from the preceding knot.</documentation>
    </annotation>
    <restriction base="string">
        <enumeration value="uniform"/>
        <enumeration value="quasiUniform"/>
        <enumeration value="piecewiseBezier"/>
    </restriction>
</simpleType>

```

```

        </restriction>
    </simpleType>
    <!-- ===== -->
</schema>
```

## C.18 gml.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:gml="http://www.opengis.net/gml" xmlns:sch="http://www.ascc.net/xml/schematron"
  xmlns="http://www.w3.org/2001/XMLSchema" xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
  attributeFormDefault="unqualified" version="3.1.0">
    <xsd:annotation>
      <xsd:appinfo source="urn:opengis:specification:gml:schema-xsd:gml:v3.1.0">gml.xsd</xsd:appinfo>
      <xsd:documentation>
        Copyright (c) 2002-2003 OGC, All Rights Reserved.
        Top level GML schema
      </xsd:documentation>
    </xsd:annotation>
    <!-- ===== -->
    <xsd:include schemaLocation="dynamicFeature.xsd"/>
    <xsd:include schemaLocation="topology.xsd"/>
    <xsd:include schemaLocation="coverage.xsd"/>
    <xsd:include schemaLocation="coordinateReferenceSystems.xsd"/>
    <xsd:include schemaLocation="observation.xsd"/>
    <xsd:include schemaLocation="defaultStyle.xsd"/>
    <xsd:include schemaLocation="temporalReferenceSystems.xsd"/>
    <!-- ===== -->
</xsd:schema>
```

## C.19 gmlBase.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:sch="http://www.ascc.net/xml/schematron" xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:gmlBase:v3.1.0">
      <sch:title>Schematron validation</sch:title>
      <sch:ns prefix="gml" uri="http://www.opengis.net/gml"/>
      <sch:ns prefix="xlink" uri="http://www.w3.org/1999/xlink"/>
      <sch:pattern name="Check either href or content not both">
        <sch:rule abstract="true" id="hrefOrContent">
          <sch:report test="@xlink:href and (*|text())">
            Property element may not carry both a reference to an object and contain an object.</sch:report>
            <sch:assert test="@xlink:href | (*|text())">
              Property element must either carry a reference to an object or contain an object.</sch:assert>
            </sch:rule>
          </sch:pattern>
        </appinfo>
        <documentation>
          GML base schema for GML 3.1
          Components to support the GML encoding model.

          The abstract Schematron rules can be used by any schema that includes gmlBase.
        </documentation>
      </annotation>
      <!-- ===== -->
      includes and imports
      <!-- ===== -->
      <include schemaLocation="basicTypes.xsd"/>
      <import namespace="http://www.w3.org/1999/xlink" schemaLocation="../xlink/xlinks.xsd"/>
      <!-- ===== -->
      <!-- ===== Objects ===== -->
      <!-- ===== -->
      <!-- ===== Abstract "Object" is "anyType" ===== -->
      <!-- ===== Global element at the head of the "Object" substitution group ===== -->
    </schema>
```

```

<element name="_Object" abstract="true">
  <annotation>
    <documentation>This abstract element is the head of a substitutionGroup hierarchy which may contain either simpleContent or complexContent elements. It is used to assert the model position of "class" elements declared in other GML schemas. </documentation>
  </annotation>
</element>
<!-- ===== Abstract "GMLObject" supertype ===== -->
<element name="_GML" type="gml:AbstractGMLType" abstract="true" substitutionGroup="gml:_Object">
  <annotation>
    <documentation>Global element which acts as the head of a substitution group that may include any element which is a GML feature, object, geometry or complex value</documentation>
  </annotation>
</element>
<!-- ===== StandardObjectProperties ===== -->
<group name="StandardObjectProperties">
  <annotation>
    <documentation>This content model group makes it easier to construct types that derive from AbstractGMLType and its descendants "by restriction". A reference to the group saves having to enumerate the standard object properties. </documentation>
  </annotation>
  <sequence>
    <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
    <element ref="gml:description" minOccurs="0"/>
    <element ref="gml:name" minOccurs="0" maxOccurs="unbounded">
      <annotation>
        <documentation>Multiple names may be provided. These will often be distinguished by being assigned by different authorities, as indicated by the value of the codeSpace attribute. In an instance document there will usually only be one name per authority. </documentation>
      </annotation>
    </element>
  </sequence>
</group>
<!-- ===== AbstractGMLType ===== -->
<complexType name="AbstractGMLType" abstract="true">
  <annotation>
    <documentation>All complexContent GML elements are directly or indirectly derived from this abstract supertype to establish a hierarchy of GML types that may be distinguished from other XML types by their ancestry. Elements in this hierarchy may have an ID and are thus referenceable. </documentation>
  </annotation>
  <sequence>
    <group ref="gml:StandardObjectProperties"/>
  </sequence>
  <attribute ref="gml:id" use="optional"/>
</complexType>
<!-- ===== Concrete "Collection" supertype ===== -->
<element name="Bag" type="gml:BagType" substitutionGroup="gml:_GML">
  <annotation>
    <documentation>Generic GML element to contain a heterogeneous collection of GML _Objects</documentation>
  </annotation>
</element>
<!-- ===== BagType ===== -->
<complexType name="BagType">
  <annotation>
    <documentation>A non-abstract generic collection type that can be used as a document element for a collection of any GML types - Geometries, Topologies, Features ...</documentation>
  </annotation>

```

"FeatureCollections" may only contain Features. "GeometryCollections" may only contain Geometry's. "Bags" are less constrained – they must contain objects that are substitutable for gml:\_Object. This may mix several levels, including Features, Definitions, Dictionaries, Geometries etc.

The content model would ideally be

- member 0..\*
- members 0..1
- member 0..\*

for maximum flexibility in building a collection from both homogeneous and distinct components:  
 included "member" elements each contain a single Object  
 an included "members" element contains a set of Objects

However, this is non-deterministic, thus prohibited by XSD.

```
</documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractGMLType">
    <sequence>
      <element ref="gml:member" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:members" minOccurs="0"/>
    </sequence>
  </extension>
</complexContent>
</complexType>
<!-- ===== -->
<!-- ===== Concrete "Array" supertype ===== -->
<element name="Array" type="gml:ArrayType" substitutionGroup="gml:_GML">
  <annotation>
    <documentation>Generic GML element to contain a homogeneous array of GML _Objects</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="ArrayType">
  <annotation>
    <documentation>A non-abstract generic collection type that can be used as a document element for a homogeneous collection of any GML types - Geometries, Topologies, Features ...</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <sequence>
        <element ref="gml:members" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== Abstract Metadata supertype ===== -->
<element name="_MetaData" type="gml:AbstractMetaDataType" abstract="true" substitutionGroup="gml:_Object">
  <annotation>
    <documentation>Abstract element which acts as the head of a substitution group for packages of MetaData properties.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractMetaDataType" abstract="true" mixed="true">
  <annotation>
    <documentation>An abstract base type for complex metadata types.</documentation>
  </annotation>
  <attribute ref="gml:id" use="optional"/>
</complexType>
<!-- ===== -->
<!-- ===== Container for Generic Metadata ===== -->
<element name="GenericMetaData" type="gml:GenericMetaDataType" substitutionGroup="gml:_MetaData">
  <annotation>
    <documentation>Concrete element in the _MetaData substitution group, which permits any well-formed XML content. Intended to act as a container for metadata defined in external schemas, for which it is not possible to add the concrete components to the GML _MetaData substitution group directly. Deprecated with GML version 3.1.0.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="GenericMetaDataType" mixed="true">
  <annotation>
    <documentation>Deprecated with GML version 3.1.0.</documentation>
  </annotation>
  <complexContent mixed="true">
    <extension base="gml:AbstractMetaDataType">
      <sequence>
        <any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

        </complexContent>
    </complexType>
    <!--
    -->
    <!-- ===== Base Property Types ===== -->
    <!--
    -->
    <!-- === property types for unspecified association - by Value or by Reference === -->
    <!-- ===== single Objects - by Value or by Reference ===== -->
    <element name="_association" type="gml:AssociationType" abstract="true"/>
    <!--
    -->
    <element name="_strictAssociation" type="gml:AssociationType" abstract="true">
        <annotation>
            <appinfo>
                <sch:pattern name="refAndContent co-occurrence prohibited">
                    <sch:rule context="gml:_strictAssociation">
                        <sch:extends rule="hrefOrContent"/>
                    </sch:rule>
                </sch:pattern>
            </appinfo>
            <documentation>must carry a reference to an object or contain an object but not both</documentation>
        </annotation>
    </element>
    <!--
    -->
    <element name="member" type="gml:AssociationType"/>
    <!--
    -->
    <complexType name="AssociationType">
        <annotation>
            <documentation> A pattern or base for derived types used to specify complex types corresponding to an unspecified UML association - either composition or aggregation. Restricts the cardinality of Objects contained in the association to a maximum of one. An instance of this type can contain an element representing an Object, or serve as a pointer to a remote Object.
        </documentation>

```

Descendents of this type can be restricted in an application schema to  
 \* allow only specified classes as valid participants in the aggregation  
 \* allow only association by reference (i.e. empty the content model) or by value (i.e. remove the xlink).

When used for association by reference, the value of the gml:remoteSchema attribute can be used to locate a schema fragment that constrains the target instance.

In many cases it is desirable to impose the constraint prohibiting the occurrence of both reference and value in the same instance, as that would be ambiguous. This is accomplished by adding a directive in the annotation element of the element declaration. This directive can be in the form of normative prose, or can use a Schematron pattern to automatically constrain co-occurrence - see the declaration for \_strictAssociation below.

If co-occurrence is not prohibited, then both a link and content may be present. If this occurs in an instance, then the rule for interpretation is that the instance found by traversing the href provides the normative value of the property, and should be used when possible. The value(s) included as content may be used if the remote instance cannot be resolved. This may be considered to be a "cached" version of the value(s). </documentation>

```

        </annotation>
        <sequence>
            <element ref="gml:_Object" minOccurs="0"/>
        </sequence>
        <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </complexType>
    <!--
    -->
    <element name="_reference" type="gml:ReferenceType" abstract="true"/>
    <!--
    -->
    <complexType name="ReferenceType">
        <annotation>
            <documentation> A pattern or base for derived types used to specify complex types corresponding to a UML aggregation association. An instance of this type serves as a pointer to a remote Object.
        </documentation>
    </annotation>
        <sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </sequence>
    </complexType>
    <!--
    -->
    <!-- ===== multiple objects - by Value or by Reference ===== -->
    <element name="members" type="gml:ArrayAssociationType"/>
    <!--
    -->
    <complexType name="ArrayAssociationType">
        <annotation>

```

<documentation> A base for derived types used to specify complex types containing an array of objects, by unspecified UML association - either composition or aggregation. An instance of this type contains elements representing Objects.

Ideally this type would be derived by extension of AssociationType.

However, this leads to a non-deterministic content model, since both the base and the extension have minOccurs="0", and is thus prohibited in XML Schema. </documentation>

```

</annotation>
<sequence>
    <element ref="gml:_Object" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
</complexType>
<!-- ===== -->
<!-- ===== Abstract "property" supertype ===== -->
<element name="metaDataProperty" type="gml:MetaData.PropertyType">
    <annotation>
        <documentation>Contains or refers to a metadata package that contains metadata properties. </documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="MetaData.PropertyType">
    <annotation>
        <documentation>Base type for complex metadata property types.</documentation>
    </annotation>
    <choice minOccurs="0">
        <element ref="gml:_MetaDataSet"/>
        <any processContents="lax"/>
    </choice>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
    <attribute name="about" type="anyURI" use="optional"/>
</complexType>
<!-- ===== -->
<!-- ===== global attribute, attribute group and element declarations
===== -->
<attribute name="id" type="ID">
    <annotation>
        <documentation>Database handle for the object. It is of XML type "ID", so is constrained to be unique in the XML document within which it occurs. An external identifier for the object in the form of a URI may be constructed using standard XML and XPointer methods. This is done by concatenating the URI for the document, a fragment separator "#", and the value of the id attribute. </documentation>
    </annotation>
</attribute>
<!-- ===== -->
<attribute name="remoteSchema" type="anyURI">
    <annotation>
        <documentation>Reference to an XML Schema fragment that specifies the content model of the property's value. This is in conformance with the XML Schema Section 4.14 Referencing Schemas from Elsewhere. </documentation>
    </annotation>
</attribute>
<!-- ===== -->
<attributeGroup name="AssociationAttributeGroup">
    <annotation>
        <documentation>Attribute group used to enable property elements to refer to their value remotely. It contains the "simple link" components from xlink.xsd, with all members "optional", and the remoteSchema attribute, which is also optional. These attributes can be attached to any element, thus allowing it to act as a pointer. The 'remoteSchema' attribute allows an element that carries link attributes to indicate that the element is declared in a remote schema rather than by the schema that constrains the current document instance. </documentation>
    </annotation>
    <attributeGroup ref="xlink:simpleLink"/>
    <attribute ref="gml:remoteSchema" use="optional"/>
</attributeGroup>
<!-- ===== -->
<element name="name" type="gml:CodeType">
    <annotation>
        <documentation>Label for the object, normally a descriptive name. An object may have several names, typically assigned by different authorities. The authority for a name is indicated by the value of its (optional) codeSpace attribute. The name may or may not be unique, as determined by the rules of the organization responsible for the codeSpace. </documentation>
    </annotation>

```

```

        </annotation>
    </element>
    <!-- ===== -->
    <element name="description" type="gml:StringOrRefType">
        <annotation>
            <documentation>Contains a simple text description of the object, or refers to an external description.</documentation>
        </annotation>
    </element>
    <!-- ===== -->
    <complexType name="StringOrRefType">
        <annotation>
            <documentation>

```

This type is available wherever there is a need for a "text" type property. It is of string type, so the text can be included inline, but the value can also be referenced remotely via xlink from the AssociationAttributeGroup. If the remote reference is present, then the value obtained by traversing the link should be used, and the string content of the element can be used for an annotation.

```

</documentation>
        </annotation>
        <simpleContent>
            <extension base="string">
                <attributeGroup ref="gml:AssociationAttributeGroup"/>
            </extension>
        </simpleContent>
    </complexType>
    <!-- ===== -->
</schema>

```

## C.20 grids.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink"
       xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified"
       version="3.1.0">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-xsd:grids:v3.1.0">grids.xsd</appinfo>
        <documentation xml:lang="en">Grid geometries
A subset of implicit geometries
Designed for use with GML Coverage schema, but maybe useful elsewhere as well.

```

Copyright (c) 2002 OGC, All Rights Reserved.

```

</documentation>
    </annotation>
    <!-- ===== -->
    includes and imports
    =====-->
    <include schemaLocation="geometryBasic0d1d.xsd"/>
    <!-- ===== -->
    global elements
    =====-->
    <element name="_ImplicitGeometry" type="gml:AbstractGeometryType" abstract="true"
substitutionGroup="gml:_Geometry"/>
    <!-- ===== -->
    <element name="Grid" type="gml:GridType" substitutionGroup="gml:_ImplicitGeometry"/>
    <!-- ===== -->
    <complexType name="GridType">
        <annotation>
            <documentation>Implicitly defines an unrectified grid, which is a network composed of two or more sets of equally
spaced parallel lines in which the members of each set intersect the members of the other sets at right angles.</documentation>
        </annotation>
        <complexContent>
            <extension base="gml:AbstractGeometryType">
                <sequence>
                    <element name="limits" type="gml:GridLimitsType"/>
                    <element name="axisName" type="string" maxOccurs="unbounded"/>
                </sequence>
                <attribute name="dimension" type="positiveInteger" use="required"/>
            </extension>
        </complexContent>
    </complexType>
    <!-- ===== -->

```

```

<complexType name="GridLimitsType">
  <sequence>
    <element name="GridEnvelope" type="gml:GridEnvelopeType"/>
  </sequence>
</complexType>
<!-- ===== -->
<complexType name="GridEnvelopeType">
  <annotation>
    <documentation>Provides grid coordinate values for the diametrically opposed corners of an envelope that bounds a section of grid. The value of a single coordinate is the number of offsets from the origin of the grid in the direction of a specific axis.</documentation>
  </annotation>
  <sequence>
    <element name="low" type="gml:integerList"/>
    <element name="high" type="gml:integerList"/>
  </sequence>
</complexType>
<!-- ===== -->
<element name="RectifiedGrid" type="gml:RectifiedGridType" substitutionGroup="gml:Grid"/>
<!-- ===== -->
<complexType name="RectifiedGridType">
  <annotation>
    <documentation>A rectified grid has an origin and vectors that define its post locations.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:GridType">
      <sequence>
        <element name="origin" type="gml:PointPropertyType"/>
        <element name="offsetVector" type="gml:VectorType" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
</schema>

```

## C.21 measures.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified"
  version="3.00" xml:lang="en">
  <xsd:annotation>
    <xsd:appinfo source="urn:opengis:specification:gml:schema-measures:v3.1.0"/>
    <xsd:documentation>
      <name>measures.xsd</name>
      <version>3.0</version>
      <scope>How to encode measures, each with associated unit of measure (uom). </scope>
      <description>Extends the units.xsd and basicTypes.xsd schemas with types for recording measures using specific types of units, especially the measures and units needed for coordinate reference systems and coordinate operations. The specific unit types encoded are length, angle, scale factor, time, area, volume, speed, and grid length. This schema allows angle values to be recorded as single numbers or in degree-minute-second format. GML 3.0 candidate schema, primary editor: Arliss Whiteside. Last updated 2002/11/13. </description>
      <copyright>Copyright (c) 2001-2002 OpenGIS, All Rights Reserved</copyright>
      <conformance>Parts of this schema are based on Subclause 6.5.7 of ISO/CD 19103 Geographic information - Conceptual schema language, on Subclause A.5.2.2.3 of ISO/CD 19118 Geographic information - Encoding, and on Subclause 4.7 of OpenGIS Recommendation Paper OGC 02-007r4 Units of Measure Use and Definition Recommendations.</conformance>
    </xsd:documentation>
  </xsd:annotation>
  <!-- ===== -->
  includes and imports
  <!-- ===== -->
  <xsd:include schemaLocation="units.xsd"/>
  <!-- ===== -->
  elements and types
  <!-- ===== -->
  <!-- This schema uses the gml:MeasureType defined in basicTypes.xsd with the modified meaning:>

```

<documentation>Value of a quantity, with its units. This element uses the XML Schema primitive data type "double" because it supports both decimal and scientific notation, and thus offers flexibility and precision. However, there is no requirement to store values using any particular format, and applications receiving elements of this type may choose to coerce the data to any other type as convenient. The XML attribute uom references the units or scale by which the amount should be multiplied. For a reference within the same XML document, the abbreviated XPointer prefix "#" symbol should be used, followed by a text abbreviation of the unit name. However, the "#" symbol may be optional, and still may be interpreted as a reference.

</documentation>-->

```

<!-- ===== -->
<xsd:element name="measure" type="gml:MeasureType"/>
<!-- ===== -->

```

<xsd:complexType name="LengthType">

<xsd:annotation>

<xsd:documentation>Value of a length (or distance) quantity, with its units. Uses the MeasureType with the restriction that the unit of measure referenced by uom must be suitable for a length, such as metres or feet. </xsd:documentation>

</xsd:annotation>

<xsd:simpleContent>

<xsd:extension base="gml:MeasureType"/>

</xsd:simpleContent>

</xsd:complexType>

<!-- ===== -->

<xsd:complexType name="ScaleType">

<xsd:annotation>

<xsd:documentation>Value of a scale factor (or ratio) that has no physical unit. Uses the MeasureType with the restriction that the unit of measure referenced by uom must be suitable for a scale factor, such as percent, permil, or parts-per-million. </xsd:documentation>

</xsd:annotation>

<xsd:simpleContent>

<xsd:extension base="gml:MeasureType"/>

</xsd:simpleContent>

</xsd:complexType>

<!-- ===== -->

<xsd:complexType name="TimeType">

<xsd:annotation>

<xsd:documentation>Value of a time or temporal quantity, with its units. Uses the MeasureType with the restriction that the unit of measure referenced by uom must be suitable for a time value, such as seconds or weeks. </xsd:documentation>

</xsd:annotation>

<xsd:simpleContent>

<xsd:extension base="gml:MeasureType"/>

</xsd:simpleContent>

</xsd:complexType>

<!-- ===== -->

<xsd:complexType name="GridLengthType">

<xsd:annotation>

<xsd:documentation>Value of a length (or distance) quantity in a grid, where the grid spacing does not have any associated physical units, or does not have a constant physical spacing. This grid length will often be used in a digital image grid, where the base units are likely to be pixel spacings. Uses the MeasureType with the restriction that the unit of measure referenced by uom must be suitable for length along the axes of a grid, such as pixel spacings or grid spacings.

</xsd:documentation>

</xsd:annotation>

<xsd:simpleContent>

<xsd:extension base="gml:MeasureType"/>

</xsd:simpleContent>

</xsd:complexType>

<!-- ===== -->

<xsd:complexType name="AreaType">

<xsd:annotation>

<xsd:documentation>Value of a spatial area quantity, with its units. Uses the MeasureType with the restriction that the unit of measure referenced by uom must be suitable for an area, such as square metres or square miles.

</xsd:documentation>

</xsd:annotation>

<xsd:simpleContent>

<xsd:extension base="gml:MeasureType"/>

</xsd:simpleContent>

</xsd:complexType>

<!-- ===== -->

<xsd:complexType name="VolumeType">

<xsd:annotation>

<xsd:documentation>Value of a spatial volume quantity, with its units. Uses the MeasureType with the restriction that the unit of measure referenced by uom must be suitable for a volume, such as cubic metres or cubic feet. </xsd:documentation>

</xsd:annotation>

<xsd:simpleContent>

```

        <xsd:extension base="gml:MeasureType"/>
    </xsd:simpleContent>
</xsd:complexType>
<!-- ===== -->
<xsd:complexType name="SpeedType">
    <xsd:annotation>
        <xsd:documentation>Value of a speed, with its units. Uses the MeasureType with the restriction that the unit of measure referenced by uom must be suitable for a velocity, such as metres per second or miles per hour.</xsd:documentation>
    </xsd:annotation>
    <xsd:simpleContent>
        <xsd:extension base="gml:MeasureType"/>
    </xsd:simpleContent>
</xsd:complexType>
<!-- ===== -->
<xsd:complexType name="AngleChoiceType">
    <xsd:annotation>
        <xsd:documentation>Value of an angle quantity provided in either degree-minute-second format or single value format. </xsd:documentation>
    </xsd:annotation>
    <xsd:choice>
        <xsd:element ref="gml:angle"/>
        <xsd:element ref="gml:dmsAngle"/>
    </xsd:choice>
</xsd:complexType>
<!-- ===== -->
<xsd:element name="angle" type="gml:AngleType"/>
<!-- ===== -->
<xsd:complexType name="AngleType">
    <xsd:annotation>
        <xsd:documentation>Value of an angle quantity recorded as a single number, with its units. Uses the MeasureType with the restriction that the unit of measure referenced by uom must be suitable for an angle, such as degrees or radians.</xsd:documentation>
    </xsd:annotation>
    <xsd:simpleContent>
        <xsd:extension base="gml:MeasureType"/>
    </xsd:simpleContent>
</xsd:complexType>
<!-- ===== -->
<xsd:element name="dmsAngle" type="gml:DMSAngleType"/>
<!-- ===== -->
<xsd:complexType name="DMSAngleType">
    <xsd:annotation>
        <xsd:documentation>Angle value provided in degree-minute-second or degree-minute format. </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:element ref="gml:degrees"/>
        <xsd:choice minOccurs="0">
            <xsd:element ref="gml:decimalMinutes"/>
            <xsd:sequence>
                <xsd:element ref="gml:minutes"/>
                <xsd:element ref="gml:seconds" minOccurs="0"/>
            </xsd:sequence>
        </xsd:choice>
    </xsd:sequence>
</xsd:complexType>
<!-- ===== -->
<xsd:element name="degrees" type="gml:DegreesType"/>
<!-- ===== -->
<xsd:complexType name="DegreesType">
    <xsd:annotation>
        <xsd:documentation>Integer number of degrees, plus the angle direction. This element can be used for geographic Latitude and Longitude. For Latitude, the XML attribute direction can take the values "N" or "S", meaning North or South of the equator. For Longitude, direction can take the values "E" or "W", meaning East or West of the prime meridian. This element can also be used for other angles. In that case, the direction can take the values "+" or "-" (of SignType), in the specified rotational direction from a specified reference direction. </xsd:documentation>
    </xsd:annotation>
    <xsd:simpleContent>
        <xsd:extension base="gml:DegreeValueType">
            <xsd:attribute name="direction">

```

```

<xsd:simpleType>
  <xsd:union>
    <xsd:simpleType>
      <xsd:restriction base="string">
        <xsd:enumeration value="N"/>
        <xsd:enumeration value="E"/>
        <xsd:enumeration value="S"/>
        <xsd:enumeration value="W"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType>
      <xsd:restriction base="gml:SignType"/>
    </xsd:simpleType>
  </xsd:union>
</xsd:simpleType>
</xsd:attribute>
</xsd:extension>
</xsd:simpleContent>
</xsd:complexType>
<!-- ===== -->
<xsd:simpleType name="DegreeValueType">
  <xsd:annotation>
    <xsd:documentation>Integer number of degrees in a degree-minute-second or degree-minute angular value, without indication of direction. </xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="nonNegativeInteger">
    <xsd:maxInclusive value="359"/>
  </xsd:restriction>
</xsd:simpleType>
<!-- ===== -->
<xsd:element name="decimalMinutes" type="gml:DecimalMinutesType"/>
<!-- ===== -->
<xsd:simpleType name="DecimalMinutesType">
  <xsd:annotation>
    <xsd:documentation>Decimal number of arc-minutes in a degree-minute angular value. </xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="decimal">
    <xsd:minInclusive value="0.00"/>
    <xsd:maxExclusive value="60.00"/>
  </xsd:restriction>
</xsd:simpleType>
<!-- ===== -->
<xsd:element name="minutes" type="gml:ArcMinutesType"/>
<!-- ===== -->
<xsd:simpleType name="ArcMinutesType">
  <xsd:annotation>
    <xsd:documentation>Integer number of arc-minutes in a degree-minute-second angular value. </xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="nonNegativeInteger">
    <xsd:maxInclusive value="59"/>
  </xsd:restriction>
</xsd:simpleType>
<!-- ===== -->
<xsd:element name="seconds" type="gml:ArcSecondsType"/>
<!-- ===== -->
<xsd:simpleType name="ArcSecondsType">
  <xsd:annotation>
    <xsd:documentation>Number of arc-seconds in a degree-minute-second angular value. </xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="decimal">
    <xsd:minInclusive value="0.00"/>
    <xsd:maxExclusive value="60.00"/>
  </xsd:restriction>
</xsd:simpleType>
<!-- ===== -->
</xsd:schema>

```

## C.22 observation.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<schema targetNamespace="http://www.opengis.net/gml" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="3.1.0_20031217">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:observation:v3.1.0">observation.xsd</appinfo>
    <documentation>Observation schema for GML 3.1
  </documentation>
  </annotation>
  <!-- ===== includes and imports ===== -->
  <!-- === properties === -->
  <include schemaLocation="feature.xsd"/>
  <include schemaLocation="direction.xsd"/>
  <include schemaLocation="valueObjects.xsd"/>
  <!-- ===== properties ===== -->
  <element name="using" type="gml:FeaturePropertyType">
    <annotation>
      <documentation>This element contains or points to a description of a sensor, instrument or procedure used for the observation</documentation>
    </annotation>
  </element>
  <!-- ===== -->
  <element name="target" type="gml:TargetPropertyType">
    <annotation>
      <documentation>This element contains or points to the specimen, region or station which is the object of the observation</documentation>
    </annotation>
  </element>
  <!-- ===== -->
  <element name="subject" type="gml:TargetPropertyType" substitutionGroup="gml:target">
    <annotation>
      <documentation>Synonym for target - common word used for photographs</documentation>
    </annotation>
  </element>
  <!-- ===== -->
  <complexType name="TargetPropertyType">
    <annotation>
      <documentation>Container for an object representing the target or subject of an observation.</documentation>
    </annotation>
    <choice minOccurs="0">
      <element ref="gml:_Feature"/>
      <element ref="gml:_Geometry"/>
    </choice>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </complexType>
  <!-- ===== -->
  <element name="resultOf" type="gml:AssociationType">
    <annotation>
      <documentation>The result of the observation: an image, external object, etc</documentation>
    </annotation>
  </element>
  <!-- ===== -->
  <!-- ===== Features ===== -->
  <element name="Observation" type="gml:ObservationType" substitutionGroup="gml:_Feature">
  <!-- ===== -->
  <complexType name="ObservationType">
    <complexContent>
      <extension base="gml:AbstractFeatureType">
        <sequence>
          <element ref="gml:validTime"/>
          <element ref="gml:using" minOccurs="0"/>
          <element ref="gml:target" minOccurs="0"/>
          <element ref="gml:resultOf"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <!-- ===== -->
  <element name="DirectedObservation" type="gml:DirectedObservationType" substitutionGroup="gml:Observation"/>
  <!-- ===== -->

```

```

<complexType name="DirectedObservationType">
  <complexContent>
    <extension base="gml:ObservationType">
      <sequence>
        <element ref="gml:direction"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="DirectedObservationAtDistance" type="gml:DirectedObservationAtDistanceType"
substitutionGroup="gml:DirectedObservation"/>
<!-- ===== -->
<complexType name="DirectedObservationAtDistanceType">
  <complexContent>
    <extension base="gml:DirectedObservationType">
      <sequence>
        <element name="distance" type="gml:MeasureType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
</schema>

```

## C.23 referenceSystems.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="3.1.0" xml:lang="en">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-referenceSystems"/>
    <documentation>
      <name>referenceSystems.xsd</name>
      <version>3.1.0</version>
      <scope>How to encode reference system definitions. </scope>
      <description>Builds on several other parts of GML 3 to encode the data needed to define reference systems. Primary editor: Arliss Whiteside. Last updated 2003/10/16. </description>
      <copyright>Copyright (c) 2002-2003 Open GIS Consortium, All Rights Reserved.</copyright>
      <conformance>This schema encodes the Reference System (RS_) package of the extended UML Model for OGC Abstract Specification Topic 2: Spatial Referencing by Coordinates. That UML model is adapted from ISO 19111 - Spatial referencing by coordinates, as described in Annex C of Topic 2. The SC_CRS class is also encoded here, to eliminate the (circular) references from coordinateOperations.xsd to coordinateReferenceSystems.xsd. The RS_SpatialReferenceSystemUsingGraphicIdentifier class is not encoded, since it is not applicable to coordinate positions. The CI_Citation class is not directly encoded, since such information can be included as metaDataProperty elements which are optionally allowed. A modified version of the EX_Extent (DataType) class from ISO 19115 is currently encoded here, using GML 3 schema types. (A more extensive version of the EX_Extent package might be XML encoded in the future, probably in a separate extent.xsd schema.) </conformance>
    </documentation>
  </annotation>
  <!-- ===== -->
  includes and imports
  =====>
  <include schemaLocation="geometryBasic2d.xsd"/>
  <include schemaLocation="temporal.xsd"/>
  <!-- ===== -->
  elements and types
  =====>
  <element name="_ReferenceSystem" type="gml:AbstractReferenceSystemType" abstract="true"
substitutionGroup="gml:Definition"/>
  <!-- ===== -->
  <complexType name="AbstractReferenceSystemBaseType" abstract="true">
    <annotation>
      <documentation>Basic encoding for reference system objects, simplifying and restricting the DefinitionType as needed. </documentation>
    </annotation>
    <complexContent>
      <restriction base="gml:DefinitionType">
        <sequence>
          <element ref="gml:metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>

```

```

        <element ref="gml:remarks" minOccurs="0">
            <annotation>
                <documentation>Comments on or information about this reference system, including source
information. </documentation>
            </annotation>
        </element>
        <element ref="gml:srsName"/>
    </sequence>
    <attribute ref="gml:id" use="required"/>
</restriction>
</complexContent>
</complexType>
<!-- ===== -->
<element name="srsName" type="gml:SimpleNameType" substitutionGroup="gml:name">
    <annotation>
        <documentation>The name by which this reference system is identified. </documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractReferenceSystemType" abstract="true">
    <annotation>
        <documentation>Description of a spatial and/or temporal reference system used by a dataset. </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractReferenceSystemBaseType">
            <sequence>
                <element ref="gml:srsID" minOccurs="0" maxOccurs="unbounded">
                    <annotation>
                        <documentation>Set of alternative identifications of this reference system. The first srsID, if any, is
normally the primary identification code, and any others are aliases. </documentation>
                    </annotation>
                </element>
                <element ref="gml:validArea" minOccurs="0"/>
                <element ref="gml:scope" minOccurs="0"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="srsID" type="gml:IdentifierType">
    <annotation>
        <documentation>An identification of a reference system. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="referenceSystemRef" type="gml:ReferenceSystemRefType" substitutionGroup="gml:dictionaryEntry"/>
<!-- ===== -->
<complexType name="ReferenceSystemRefType">
    <annotation>
        <documentation>Association to a reference system, either referencing or containing the definition of that reference
system. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:DictionaryEntryType">
            <sequence>
                <element ref="gml:_ReferenceSystem" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<element name="_CRS" type="gml:AbstractCRSType" abstract="true" substitutionGroup="gml:_ReferenceSystem"/>
<!-- ===== -->
<complexType name="AbstractCRSType" abstract="true">
    <annotation>
        <documentation>Abstract coordinate reference system, usually defined by a coordinate system and a datum. This
abstract complexType shall not be used, extended, or restricted, in an Application Schema, to define a concrete subtype with a
meaning equivalent to a concrete subtype specified in this document. </documentation>
    </annotation>

```

```

</annotation>
<complexContent>
    <extension base="gml:AbstractReferenceSystemType"/>
</complexContent>
</complexType>
<!-- ===== -->
<element name="crsRef" type="gml:CRSRefType" substitutionGroup="gml:referenceSystemRef"/>
<!-- ===== -->
<complexType name="CRSRefType">
    <annotation>
        <documentation>Association to a CRS abstract coordinate reference system, either referencing or containing the definition of that CRS. </documentation>
    </annotation>
    <complexContent>
        <restriction base="gml:ReferenceSystemRefType">
            <sequence>
                <element ref="gml:_CRS" minOccurs="0"/>
            </sequence>
            <attributeGroup ref="gml:AssociationAttributeGroup"/>
        </restriction>
    </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<complexType name="SimpleNameType">
    <annotation>
        <documentation>The primary name of a reference system object. The string in the CodeType contains the object identification name, and the codeSpace attribute is not included. </documentation>
    </annotation>
    <simpleContent>
        <restriction base="gml:CodeType">
            <attribute name="codeSpace" type="anyURI" use="prohibited"/>
        </restriction>
    </simpleContent>
</complexType>
<!-- ===== -->
<complexType name="IdentifierType">
    <annotation>
        <documentation>An identification of a CRS object. The first use of the IdentifierType for an object, if any, is normally the primary identification code, and any others are aliases. </documentation>
    </annotation>
    <sequence>
        <element ref="gml:name">
            <annotation>
                <documentation>The code or name for this Identifier, often from a controlled list or pattern defined by a code space. The optional codeSpace attribute is normally included to identify or reference a code space within which one or more codes are defined. This code space is often defined by some authority organization, where one organization may define multiple code spaces. The range and format of each Code Space identifier is defined by that code space authority. Information about that code space authority can be included as metaDataProperty elements which are optionally allowed in all CRS objects. </documentation>
            </annotation>
        </element>
        <element ref="gml:version" minOccurs="0"/>
        <element ref="gml:remarks" minOccurs="0">
            <annotation>
                <documentation>Remarks about this code or alias. </documentation>
            </annotation>
        </element>
    </sequence>
</complexType>
<!-- ===== -->
<element name="version" type="string">
    <annotation>
        <documentation>Identifier of the version of the associated codeSpace or code, as specified by the codeSpace or code authority. This version is included only when the "code" or "codeSpace" uses versions. When appropriate, the version is identified by the effective date, coded using ISO 8601 date format. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="remarks" type="gml:StringOrRefType" substitutionGroup="gml:description">
    <annotation>

```

```

        <documentation>Information about this object or code. Contains text or refers to external text. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="scope" type="string">
    <annotation>
        <documentation>Description of domain of usage, or limitations of usage, for which this CRS object is valid. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="validArea" type="gml:ExtentType">
    <annotation>
        <documentation>Area or region in which this CRS object is valid. </documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="ExtentType">
    <annotation>
        <documentation>Information about the spatial, vertical, and/or temporal extent of a reference system object. </documentation>
    Constraints: At least one of the elements "description", "boundingBox", "boundingPolygon", "verticalExtent", and temporalExtent" must be included, but more than one can be included when appropriate. Furthermore, more than one "boundingBox", "boundingPolygon", "verticalExtent", and/or temporalExtent" element can be included, with more than one meaning the union of the individual domains. </documentation>
        </annotation>
    <sequence>
        <element ref="gml:description" minOccurs="0">
            <annotation>
                <documentation>Description of spatial and/or temporal extent of this object. </documentation>
            </annotation>
        </element>
        <choice>
            <annotation>
                <documentation>Geographic domain of this reference system object. </documentation>
            </annotation>
            <element ref="gml:boundingBox" minOccurs="0" maxOccurs="unbounded">
                <annotation>
                    <documentation>Unordered list of bounding boxes (or envelopes) whose union describes the spatial domain of this object. </documentation>
                </annotation>
            </element>
            <element ref="gml:boundingPolygon" minOccurs="0" maxOccurs="unbounded">
                <annotation>
                    <documentation>Unordered list of bounding polygons whose union describes the spatial domain of this object. </documentation>
                </annotation>
            </element>
        </choice>
        <element ref="gml:verticalExtent" minOccurs="0" maxOccurs="unbounded">
            <annotation>
                <documentation>Unordered list of vertical intervals whose union describes the spatial domain of this object. </documentation>
            </annotation>
        </element>
        <element ref="gml:temporalExtent" minOccurs="0" maxOccurs="unbounded">
            <annotation>
                <documentation>Unordered list of time periods whose union describes the spatial domain of this object. </documentation>
            </annotation>
        </element>
    </sequence>
</complexType>
<!-- ===== -->
<element name="boundingBox" type="gml:EnvelopeType">
    <annotation>
        <documentation>A bounding box (or envelope) defining the spatial domain of this object. </documentation>
    </annotation>
</element>
<!-- ===== -->

```

```

<element name="boundingPolygon" type="gml:PolygonType">
  <annotation>
    <documentation>A bounding polygon defining the horizontal spatial domain of this object. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="verticalExtent" type="gml:EnvelopeType">
  <annotation>
    <documentation>An interval defining the vertical spatial domain of this object. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="temporalExtent" type="gml:TimePeriodType">
  <annotation>
    <documentation>A time period defining the temporal domain of this object. </documentation>
  </annotation>
</element>
<!-- ===== -->
</schema>

```

## C.24 temporal.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
       xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:temporal:v3.1.0"/>
    <documentation xml:lang="en">
      The temporal schema for GML 3.1 provides constructs for handling time-varying spatial data.
      This schema reflects a partial implementation of the model described in ISO 19108:2002.
      Copyright (c) 2004 OGC, All Rights Reserved.
    </documentation>
    </annotation>
  <!-- ===== -->
  <include schemaLocation="gmlBase.xsd"/>
  <!-- ===== -->
  <!-- ===== Time Object ===== -->
  <!-- ===== -->
  <element name="_TimeObject" type="gml:AbstractTimeObjectType" abstract="true" substitutionGroup="gml:_GML">
    <annotation>
      <documentation xml:lang="en">
        This abstract element acts as the head of the substitution group for temporal primitives and complexes.
      </documentation>
      </annotation>
    </element>
  <!-- ===== -->
  <complexType name="AbstractTimeObjectType" abstract="true">
    <annotation>
      <documentation xml:lang="en">
        The abstract supertype for temporal objects.
      </documentation>
      </annotation>
      <complexContent>
        <extension base="gml:AbstractGMLType"/>
      </complexContent>
    </complexType>
  <!-- ===== -->
  <!-- ===== Time Primitive ===== -->
  <!-- ===== -->
  <element name="_TimePrimitive" type="gml:AbstractTimePrimitiveType" abstract="true"
         substitutionGroup="gml:_TimeObject">
    <annotation>
      <documentation xml:lang="en">
        This abstract element acts as the head of the substitution group for temporal primitives.
      </documentation>
      </annotation>
    </element>
  <!-- ===== -->
  <complexType name="AbstractTimePrimitiveType" abstract="true">
    <annotation>

```

```

<documentation xml:lang="en">
The abstract supertype for temporal primitives.
</documentation>
</annotation>
<complexContent>
<extension base="gml:AbstractTimeObjectType">
<sequence>
<element name="relatedTime" type="gml:RelatedTimeType" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimePrimitivePropertyType">
<sequence minOccurs="0">
<element ref="gml:_TimePrimitive"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<complexType name="RelatedTimeType">
<complexContent>
<extension base="gml:TimePrimitivePropertyType">
<attribute name="relativePosition">
<simpleType>
<restriction base="string">
<enumeration value="Before"/>
<enumeration value="After"/>
<enumeration value="Begins"/>
<enumeration value="Ends"/>
<enumeration value="During"/>
<enumeration value="Equals"/>
<enumeration value="Contains"/>
<enumeration value="Overlaps"/>
<enumeration value="Meets"/>
<enumeration value="OverlappedBy"/>
<enumeration value="MetBy"/>
<enumeration value="BegunBy"/>
<enumeration value="EndedBy"/>
</restriction>
</simpleType>
</attribute>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<!-- == Time Complex == -->
<!-- ===== -->
<element name="_TimeComplex" type="gml:AbstractTimeComplexType" abstract="true"
substitutionGroup="gml:_TimeObject">
<annotation>
<documentation xml:lang="en">
This abstract element acts as the head of the substitution group for temporal complexes.
Temporal complex is an aggregation of temporal primitives as its components,
represents a temporal geometric complex and a temporal topology complex.
N.B. Temporal geometric complex is not defined in this schema.
</documentation>
</annotation>
</element>
<!-- ===== -->
<complexType name="AbstractTimeComplexType" abstract="true">
<annotation>
<documentation xml:lang="en">
The abstract supertype for temporal complexes.
</documentation>
</annotation>
<complexContent>
<extension base="gml:AbstractTimeObjectType"/>
</complexContent>

```

```

</complexType>
<!-- ===== Time Geometric Primitive ===== -->
<!-- ===== Time Geometric Primitive ===== -->
<element name="_TimeGeometricPrimitive" type="gml:AbstractTimeGeometricPrimitiveType" abstract="true"
substitutionGroup="gml:_TimePrimitive">
  <annotation>
    <documentation xml:lang="en">
      This abstract element acts as the head of the substitution group for temporal geometric primitives.
    </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractTimeGeometricPrimitiveType" abstract="true">
  <annotation>
    <documentation xml:lang="en">
      The abstract supertype for temporal geometric primitives.
      A temporal geometry must be associated with a temporal reference system via URI.
      The Gregorian calendar with UTC is the default reference system, following ISO
      8601. Other reference systems in common use include the GPS calendar and the
      Julian calendar.
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTimePrimitiveType">
      <attribute name="frame" type="anyURI" use="optional" default="#ISO-8601"/>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimeGeometricPrimitivePropertyType">
  <sequence minOccurs="0">
    <element ref="gml:_TimeGeometricPrimitive"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== Time Instant ===== -->
<!-- ===== -->
<element name="TimeInstant" type="gml:TimeInstantType" substitutionGroup="gml:_TimeGeometricPrimitive"/>
<!-- ===== -->
<complexType name="TimeInstantType">
  <annotation>
    <documentation>Omit back-pointers begunBy, endedBy. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTimeGeometricPrimitiveType">
      <sequence>
        <element ref="gml:timePosition"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimeInstantPropertyType">
  <sequence minOccurs="0">
    <element ref="gml:TimeInstant"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== Time Period ===== -->
<!-- ===== -->
<element name="TimePeriod" type="gml:TimePeriodType" substitutionGroup="gml:_TimeGeometricPrimitive"/>
<!-- ===== -->
<complexType name="TimePeriodType">
  <complexContent>
    <extension base="gml:AbstractTimeGeometricPrimitiveType">
      <sequence>
        <choice>
          <element name="beginPosition" type="gml:TimePositionType"/>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

        <element name="begin" type="gml:TimeInstantPropertyType"/>
    </choice>
    <choice>
        <element name="endPosition" type="gml:TimePositionType"/>
        <element name="end" type="gml:TimeInstantPropertyType"/>
    </choice>
    <element ref="gml:_timeLength" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimePeriodPropertyType">
    <sequence minOccurs="0">
        <element ref="gml:TimePeriod"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- ===== duration & interval ===== -->
<!-- ===== -->
<!-- ===== -->
<element name="_timeLength" type="gml:TimeLengthType" abstract="true">
    <annotation>
        <documentation xml:lang="en">
This abstract element serves as the head of the substitution group for elements used
to indicate temporal length or distance (duration, interval).
</documentation>
    </annotation>
</element>
<!-- ===== -->
<simpleType name="TimeLengthType">
    <annotation>
        <documentation xml:lang="en">
Base type for describing temporal length or distance. The value space is further
constrained by subtypes that conform to the ISO 8601 or ISO 11404 standards.
</documentation>
    </annotation>
    <union memberTypes="duration decimal"/>
</simpleType>
<!-- ===== -->
<element name="duration" type="duration" substitutionGroup="gml:_timeLength">
    <annotation>
        <documentation xml:lang="en">
This element is an instance of the primitive xsd:duration simple type to
enable use of the ISO 8601 syntax for temporal length (e.g. P5DT4H30M).
It is a valid subtype of TimeDurationType according to section 3.14.6,
rule 2.2.4 in XML Schema, Part 1.
</documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="timeInterval" type="gml:TimeIntervalLengthType" substitutionGroup="gml:_timeLength">
    <annotation>
        <documentation>
This element is a valid subtype of TimeDurationType
according to section 3.14.6, rule 2.2.4 in XML Schema, Part 1.
</documentation>
    </annotation>
</element>
<!-- ===== -->
<complexType name="TimeIntervalLengthType" final="#all">
    <annotation>
        <documentation xml:lang="en">
This type extends the built-in xsd:decimal simple type to allow floating-point
values for temporal length. According to the ISO 11404 model you have to use
positiveInteger together with appropriate values for radix and factor. The
resolution of the time interval is to one radix ^(-factor) of the specified
time unit (e.g. unit="second", radix="10", factor="3" specifies a resolution
of milliseconds). It is a subtype of TimeDurationType.

```

```

</documentation>
</annotation>
<simpleContent>
  <extension base="decimal">
    <attribute name="unit" type="gml:TimeUnitType" use="required"/>
    <attribute name="radix" type="positiveInteger" use="optional"/>
    <attribute name="factor" type="integer" use="optional"/>
  </extension>
</simpleContent>
</complexType>
<!-- ===== -->
<simpleType name="TimeUnitType">
  <annotation>
    <documentation xml:lang="en">
      Standard units for measuring time intervals (see ISO 31-1).
    </documentation>
  </annotation>
  <union>
    <simpleType>
      <restriction base="string">
        <enumeration value="year"/>
        <enumeration value="day"/>
        <enumeration value="hour"/>
        <enumeration value="minute"/>
        <enumeration value="second"/>
      </restriction>
    </simpleType>
    <simpleType>
      <restriction base="string">
        <pattern value="other:\w{2,}" />
      </restriction>
    </simpleType>
  </union>
</simpleType>
<!-- ===== -->
<!-- Time Position ===== -->
<!-- ===== -->
<element name="timePosition" type="gml:TimePositionType">
  <annotation>
    <documentation>Direct representation of a temporal position</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="TimePositionType" final="#all">
  <annotation>
    <documentation xml:lang="en">Direct representation of a temporal position.
    Indeterminate time values are also allowed, as described in ISO 19108. The indeterminatePosition
    attribute can be used alone or it can qualify a specific value for temporal position (e.g. before
    2002-12, after 1019624400).
    For time values that identify position within a calendar, the calendarEraName attribute provides
    the name of the calendar era to which the date is referenced (e.g. the Meiji era of the Japanese calendar).
  </annotation>
  <simpleContent>
    <extension base="gml:TimePositionUnion">
      <attribute name="frame" type="anyURI" use="optional" default="#ISO-8601"/>
      <attribute name="calendarEraName" type="string" use="optional"/>
      <attribute name="indeterminatePosition" type="gml:TimeIndeterminateValueType" use="optional"/>
    </extension>
  </simpleContent>
</complexType>
<!-- ===== -->
<simpleType name="TimePositionUnion">
  <annotation>
    <documentation xml:lang="en">
      The ISO 19108:2002 hierarchy of subtypes for temporal position are collapsed
      by defining a union of XML Schema simple types for indicating temporal position relative
      to a specific reference system.
    </documentation>
  </annotation>

```

Dates and dateTime may be indicated with varying degrees of precision.  
 dateTime by itself does not allow right-truncation, except for fractions of seconds.

When used with non-Gregorian calendars based on years, months, days, the same lexical representation should still be used, with leading zeros added if the year value would otherwise have fewer than four digits.

An ordinal position may be referenced via URI identifying the definition of an ordinal era.

A time coordinate value is indicated as a decimal (e.g. UNIX time, GPS calendar).

```
</documentation>
</annotation>
<union memberTypes="gml:CalDate time dateTime anyURI decimal"/>
</simpleType>
<!-- ===== -->
<simpleType name="CalDate">
<annotation>
<documentation xml:lang="en">
```

Calendar dates may be indicated with varying degrees of precision, using year, year-month, date.

When used with non-Gregorian calendars based on years, months, days, the same lexical representation should still be used, with leading zeros added if the year value would otherwise have fewer than four digits.

time is used for a position that recurs daily (see clause 5.4.4.2 of ISO 19108:2002).

```
</documentation>
</annotation>
<union memberTypes="date gYearMonth gYear"/>
</simpleType>
<!-- ===== -->
<simpleType name="TimeIndeterminateValueType">
<annotation>
<documentation xml:lang="en">
```

This enumerated data type specifies values for indeterminate positions.

```
</documentation>
</annotation>
<restriction base="string">
<enumeration value="after"/>
<enumeration value="before"/>
<enumeration value="now"/>
<enumeration value="unknown"/>
</restriction>
</simpleType>
<!-- ===== -->
<!-- === Convenience properties === -->
<!-- ===== -->
<element name="validTime" type="gml:TimePrimitivePropertyType"/>
<!-- ===== -->
</schema>
```

## C.25 temporalReferenceSystems.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns:gml="http://www.opengis.net/gml"
       xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:temporalReferenceSystems:v3.1.0"/>
    <documentation xml:lang="en">
```

The Temporal Reference Systems schema for GML 3.1 provides constructs for handling various styles of temporal reference system.

This schema reflects a partial implementation of the model described in ISO 19108:2002.

Copyright (c) 2004 OGC, All Rights Reserved.

```
</documentation>
</annotation>
<!-- ===== -->
<include schemaLocation="temporalTopology.xsd"/>
<include schemaLocation="dictionary.xsd"/>
<!-- ===== -->
<!-- == Time Reference System == -->
<!-- ===== -->
<element name="_TimeReferenceSystem" type="gml:AbstractTimeReferenceSystemType" abstract="true"
       substitutionGroup="gml:_GML">
```

```

<annotation>
  <documentation>Abstract element serves primarily as the head of a substitution group for temporal reference systems.</documentation>
</annotation>
</element>
<!-- ===== -->
<complexType name="AbstractTimeReferenceSystemType" abstract="true">
  <annotation>
    <documentation xml:lang="en">
      A value in the time domain is measured relative to a temporal reference system. Common types of reference systems include calendars, ordinal temporal reference systems, and temporal coordinate systems (time elapsed since some epoch, e.g. UNIX time).
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:DefinitionType">
      <sequence>
        <element name="domainOfValidity" type="string" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<!-- == Time Coordinate System == -->
<!-- ===== -->
<element name="TimeCoordinateSystem" type="gml:TimeCoordinateSystemType"
substitutionGroup="gml:_TimeReferenceSystem"/>
<!-- ===== -->
<complexType name="TimeCoordinateSystemType">
  <annotation>
    <documentation xml:lang="en">
      A temporal coordinate system is based on a continuous interval scale defined in terms of a single time interval.
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTimeReferenceSystemType">
      <sequence>
        <choice>
          <element name="originPosition" type="gml:TimePositionType"/>
          <element name="origin" type="gml:TimeInstantPropertyType"/>
        </choice>
        <element name="interval" type="gml:TimeIntervalLengthType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<!-- == Time Ordinal System == -->
<!-- ===== -->
<element name="TimeOrdinalReferenceSystem" type="gml:TimeOrdinalReferenceSystemType"
substitutionGroup="gml:_TimeReferenceSystem"/>
<!-- ===== -->
<complexType name="TimeOrdinalReferenceSystemType">
  <annotation>
    <documentation xml:lang="en">
      In an ordinal reference system the order of events in time can be well established, but the magnitude of the intervals between them can not be accurately determined (e.g. a stratigraphic sequence).
    </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTimeReferenceSystemType">
      <sequence>
        <element name="component" type="gml:TimeOrdinalEraPropertyType" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="TimeOrdinalEra" type="gml:TimeOrdinalEraType"/>
<!-- ===== -->

```

```

<complexType name="TimeOrdinalEraType">
  <annotation>
    <documentation xml:lang="en">
```

Ordinal temporal reference systems are often hierarchically structured such that an ordinal era at a given level of the hierarchy includes a sequence of shorter, coterminous ordinal eras. This is captured using the member/group properties.

Note that in this schema, Time Ordinal Era is patterned on TimeEdge, which is a variation from ISO 19108. This is in order to fulfill the requirements of ordinal reference systems based on eras delimited by named points or nodes, which are common in geology, archeology, etc.

This change is subject of a change proposal to ISO

```

</annotation>
  </annotation>
  <complexContent>
    <extension base="gml:DefinitionType">
      <sequence>
        <element name="relatedTime" type="gml:RelatedTimeType" minOccurs="0" maxOccurs="unbounded"/>
        <element name="start" type="gml:TimeNodePropertyType"/>
        <element name="end" type="gml:TimeNodePropertyType"/>
        <element name="extent" type="gml:TimePeriodPropertyType" minOccurs="0"/>
        <element name="member" type="gml:TimeOrdinalEraPropertyType" minOccurs="0"
maxOccurs="unbounded">
          <annotation>
            <documentation>An Era may be composed of several member Eras. The "member" element implements the association to the Era at the next level down the hierarchy. "member" follows the standard GML property pattern whereby its (complex) value may be either described fully inline, or may be the target of a link carried on the member element and described fully elsewhere, either in the same document or from another service. </documentation>
          </annotation>
        </element>
        <element name="group" type="gml:ReferenceType" minOccurs="0">
          <annotation>
            <documentation>In a particular Time System, an Era may be a member of a group. The "group" element implements the back-pointer to the Era at the next level up in the hierarchy.

```

If the hierarchy is represented by describing the nested components fully in their nested position inside "member" elements, then the parent can be easily inferred, so the group property is unnecessary.

However, if the hierarchy is represented by links carried on the "member" property elements, pointing to Eras described fully elsewhere, then it may be useful for a child (member) era to carry an explicit pointer back to its parent (group) Era.

```

</annotation>
      </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimeOrdinalEraPropertyType">
  <sequence minOccurs="0">
    <element ref="gml:TimeOrdinalEra"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- == Calendar == -->
<!-- ===== -->
<element name="TimeCalendar" type="gml:TimeCalendarType" substitutionGroup="gml:_TimeReferenceSystem"/>
<!-- ===== -->
<complexType name="TimeCalendarType">
  <annotation>
    <documentation xml:lang="en">A calendar is a discrete temporal reference system that provides a basis for defining temporal position to a resolution of one day. A single calendar may reference more than one calendar era.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTimeReferenceSystemType">
      <sequence>
        <element name="referenceFrame" type="gml:TimeCalendarEraPropertyType" maxOccurs="unbounded">
```

```

        <annotation>
            <documentation>Link to the CalendarEras that it uses as a reference for dating.</documentation>
        </annotation>
    </element>
</sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimeCalendarPropertyType">
    <sequence minOccurs="0">
        <element ref="gml:TimeCalendar"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<element name="TimeCalendarEra" type="gml:TimeCalendarEraType"/>
<!-- ===== -->
<complexType name="TimeCalendarEraType">
    <annotation>
        <documentation xml:lang="en">
In every calendar, years are numbered relative to the date of a
reference event that defines a calendar era.
In this implementation, we omit the back-pointer "datingSystem". </documentation>
        </annotation>
        <complexContent>
            <extension base="gml:DefinitionType">
                <sequence>
                    <element name="referenceEvent" type="gml:StringOrRefType">
                        <annotation>
                            <documentation>Name or description of a mythical or historic event which fixes the position of the
base scale of the calendar era.</documentation>
                        </annotation>
                    </element>
                    <element name="referenceDate" type="date" default="0001-01-01" minOccurs="0">
                        <annotation>
                            <documentation>Date of the referenceEvent expressed as a date in the given calendar.
In most calendars, this date is the origin (i.e., the first day) of the scale, but this is not always true.</documentation>
                        </annotation>
                    </element>
                    <element name="julianReference" type="decimal">
                        <annotation>
                            <documentation>Julian date that corresponds to the reference date.
The Julian day numbering system is a temporal coordinate system that has an
origin earlier than any known calendar,
at noon on 1 January 4713 BC in the Julian proleptic calendar.
The Julian day number is an integer value;
the Julian date is a decimal value that allows greater resolution.
Transforming calendar dates to and from Julian dates provides a
relatively simple basis for transforming dates from one calendar to another.</documentation>
                        </annotation>
                    </element>
                    <element name="epochOfUse" type="gml:TimePeriodPropertyType">
                        <annotation>
                            <documentation>Period for which the calendar era was used as a basis for dating.</documentation>
                        </annotation>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
<!-- ===== -->
<complexType name="TimeCalendarEraPropertyType">
    <sequence minOccurs="0">
        <element ref="gml:TimeCalendarEra"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- Clock == -->
<!-- ===== -->

```

```

<element name="TimeClock" type="gml:TimeClockType" substitutionGroup="gml:_TimeReferenceSystem"/>
<!-- ===== -->
<complexType name="TimeClockType" final="#all">
  <annotation>
    <documentation xml:lang="en">A clock provides a basis for defining temporal position within a day.  

    A clock must be used with a calendar in order to provide a complete description of a temporal position  

    within a specific day.  </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTimeReferenceSystemType">
      <sequence>
        <element name="referenceEvent" type="gml:StringOrRefType">
          <annotation>
            <documentation>Name or description of an event, such as solar noon or sunrise,  

            which fixes the position of the base scale of the clock.</documentation>
          </annotation>
        </element>
        <element name="referenceTime" type="time">
          <annotation>
            <documentation>time of day associated with the reference event expressed as  

            a time of day in the given clock. The reference time is usually the origin of the clock scale. </documentation>
          </annotation>
        </element>
        <element name="utcReference" type="time">
          <annotation>
            <documentation>24 hour local or UTC time that corresponds to the reference time.</documentation>
          </annotation>
        </element>
        <element name="dateBasis" type="gml:TimeCalendarPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimeClock.PropertyType">
  <sequence minOccurs="0">
    <element ref="gml:TimeClock"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
</schema>

```

## C.26 temporalTopology.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns:gml="http://www.opengis.net/gml"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:temporalTopology:v3.1.0"/>
    <documentation xml:lang="en">
      The temporal topology schema for ISO19136 provides constructs for handling topological complexes and  

      temporal feature relationships.  

      Temporal geometric characteristics of features are represented as instants and periods.  

      While, temporal context of features that does not relate to the position of time is described as connectivity relationships  

      among instants and periods. These relationships are called temporal topology as they do not change in time,  

      as long as the direction of time does not change.  

      It is used effectively in the case of describing a family tree expressing evolution of species, an ecological cycle,  

      a lineage of lands or buildings, or a history of separation and merger of administrative boundaries.  

      This schema reflects a partial yet consistent implementation of the model described in ISO 19108:2002.  

      Copyright (c) 2004 OGC, All Rights Reserved.
    </documentation>
    </annotation>
  <!-- ===== -->
  <include schemaLocation="temporal.xsd"/>
  <!-- ===== -->

```

```

<!-- ===== -->
<!-- == TimeTopologyComplex == -->
<!-- ===== -->
<element name="TimeTopologyComplex" type="gml:TimeTopologyComplexType" substitutionGroup="gml:_TimeComplex">
  <annotation>
    <documentation xml:lang="en">
      This element represents temporal topology complex. It shall be the connected acyclic directed graph composed of time nodes
      and time edges.
    </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="TimeTopologyComplexType">
  <annotation>
    <documentation xml:lang="en">A temporal topology complex.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTimeComplexType">
      <sequence>
        <element name="primitive" type="gml:TimeTopologyPrimitivePropertyType" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimeTopologyComplexPropertyType">
  <annotation>
    <documentation>A time topology complex property can either be any time topology complex element
      encapsulated in an element of this type or an XLink reference to a remote time topology complex element
      (where remote includes elements located elsewhere in the same document).
      Note that either the reference or the contained element must be given, but not both or none.
    </documentation>
  </annotation>
  <sequence>
    <element ref="gml:TimeTopologyComplex" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- == TimeTopologyPrimitive == -->
<!-- ===== -->
<element name="_TimeTopologyPrimitive" type="gml:AbstractTimeTopologyPrimitiveType" abstract="true"
substitutionGroup="gml:_TimePrimitive">
  <annotation>
    <documentation xml:lang="en">
      This abstract element acts as the head of the substitution group for temporal topology primitives.
    </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="AbstractTimeTopologyPrimitiveType" abstract="true">
  <annotation>
    <documentation xml:lang="en">The element "complex" carries a reference to the complex containing this
      primitive.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTimePrimitiveType">
      <sequence>
        <element name="complex" type="gml:ReferenceType" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimeTopologyPrimitivePropertyType">
  <annotation>
    <documentation>A time topology primitive property can either hold any time topology complex element
      or carry an XLink reference to a remote time topology complex element
      (where remote includes elements located elsewhere in the same document).
      Note that either the reference or the contained element must be given, but not both or none.
    </documentation>
  </annotation>

```

```

</annotation>
<sequence>
    <element ref="gml:_TimeTopologyPrimitive" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== TimeNode ===== -->
<!-- ===== TimeNode ===== -->
<!-- ===== TimeNode ===== -->
<element name="TimeNode" type="gml:TimeNodeType" substitutionGroup="gml:_TimeTopologyPrimitive">
    <annotation>
        <documentation xml:lang="en">"TimeNode" is a zero dimensional temporal topology primitive,  

        expresses a position in topological time, and is a start and an end of time edge, which represents states of time.  

        Time node may be isolated. However, it cannot describe the ordering relationships with other primitives.  

        An isolated node may not be an element of any temporal topology complex.
        </documentation>
    </annotation>
</element>
<!-- ===== TimeNode ===== -->
<complexType name="TimeNodeType">
    <annotation>
        <documentation xml:lang="en">Type declaration of the element "TimeNode".
    </documentation>
    <annotation>
        <documentation xml:lang="en">A time node property can either be any time node element encapsulated in an element of this type  

        or an XLink reference to a remote time node element (where remote includes elements located elsewhere in the  

        same document).
        Note that either the reference or the contained element must be given, but not both or none.
        </documentation>
    </annotation>
    <sequence>
        <element ref="gml:TimeNode" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== TimeEdge ===== -->
<!-- ===== TimeEdge ===== -->
<!-- ===== TimeEdge ===== -->
<element name="TimeEdge" type="gml:TimeEdgeType" substitutionGroup="gml:_TimeTopologyPrimitive">
    <annotation>
        <documentation xml:lang="en">TimeEdge is one dimensional temporal topology primitive,  

        expresses a state in topological time. It has an orientation from its start toward the end,  

        and its boundaries shall associate with two different time nodes.
        </documentation>
    </annotation>
</element>
<!-- ===== TimeEdge ===== -->
<complexType name="TimeEdgeType">
    <annotation>
        <documentation xml:lang="en">Type declaration of the element "TimeEdge".
        </documentation>
    </annotation>
    <complexContent>
        <extension base="gml:AbstractTimeTopologyPrimitiveType">
            <sequence>

```

```

<element name="start" type="gml:TimeNodePropertyType"/>
<element name="end" type="gml:TimeNodePropertyType"/>
<element name="extent" type="gml:TimePeriodPropertyType" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== -->
<complexType name="TimeEdge.PropertyType">
<annotation>
    <documentation>A time edge property can either be any time edge element encapsulated in an element of this type or an XLink reference to a remote time edge element (where remote includes elements located elsewhere in the same document).</documentation>
    Note that either the reference or the contained element must be given, but not both or none.
</annotation>
<sequence>
    <element ref="gml:TimeEdge" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- == Succession == -->
<!-- ===== -->
<simpleType name="SuccessionType">
<annotation>
    <documentation>Feature succession is a semantic relationship derived from evaluation of observer, and "Feature Substitution", "Feature Division" and "Feature Fusion" are defined as associations between previous features and next features in the temporal context. Successions shall be represented in either following two ways.
        * define a temporal topological complex element as a feature element
        * define an association same as temporal topological complex between features.</documentation>
</annotation>
<restriction base="string">
    <enumeration value="substitution"/>
    <enumeration value="division"/>
    <enumeration value="fusion"/>
    <enumeration value="initiation"/>
</restriction>
</simpleType>
<!-- ===== -->
</schema>

```

## C.27 topology.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink"
       xmlns:gml="http://www.opengis.net/gml" xmlns:sch="http://www.ascc.net/xml/schematron"
       xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="3.1.0">
    <annotation>
        <appinfo source="urn:opengis:specification:gml:schema-xsd:topology:v3.1.0">topology.xsd</appinfo>
        <documentation>
            <include schemaLocation="geometryComplexes.xsd"/>
        <!-- ===== -->
        abstract supertype for topology objects
        <!-- ===== -->
        <!-- ===== -->
        <element name="_Topology" type="gml:AbstractTopologyType" abstract="true" substitutionGroup="gml:_Object"/>
        <!-- ===== -->
        <complexType name="AbstractTopologyType" abstract="true">
            <complexContent>
                <extension base="gml:AbstractGMLType"/>
            </complexContent>
        </complexType>
        <!-- ===== -->
        <element name="_TopoPrimitive" type="gml:AbstractTopoPrimitiveType" abstract="true"
               substitutionGroup="gml:_Topology">

```

```

<annotation>
  <documentation>Substitution group branch for Topo Primitives, used by
TopoPrimitiveArrayAssociationType</documentation>
</annotation>
</element>
<!-- ===== -->
<complexType name="AbstractTopologyType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractTopologyType">
      <sequence>
        <element ref="gml:isolated" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:container" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="isolated" type="gml:IsolatedPropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:isolated">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
  </annotation>
</element>
<!-- ===== -->
<complexType name="IsolatedPropertyType">
  <choice minOccurs="0">
    <element ref="gml:Node"/>
    <element ref="gml:Edge"/>
  </choice>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<element name="container" type="gml:ContainerPropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:containerProperty">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
  </annotation>
</element>
<!-- ===== -->
<complexType name="ContainerPropertyType">
  <choice minOccurs="0">
    <element ref="gml:Face"/>
    <element ref="gml:TopoSolid"/>
  </choice>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- primitive topology objects -->
<!-- ===== -->
<element name="Node" type="gml:NodeType" substitutionGroup="gml:_TopoPrimitive"/>
<!-- ===== -->
<complexType name="NodeType">
  <annotation>
    <documentation> Its optional co-boundary is a set of connected directedEdges. The orientation of one of these
dirEdges is "+" if the Node is the "to" node of the Edge, and "-" if it is the "from" node. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTopologyType">
      <sequence>

```

```

<element ref="gml:directedEdge" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:pointProperty" minOccurs="0"/>
<!-- <element name="geometry" type="gml:PointPropertyType" minOccurs="0"/> -->
</sequence>
</extension>
</complexContent>
</complexType>
<!-- ===== Property for topology association - by Value or by Reference ===== -->
<element name="directedNode" type="gml:DirectedNodePropertyType">
<annotation>
  <appinfo>
    <sch:pattern>
      <sch:rule context="gml:directedNode">
        <sch:extends rule="hrefOrContent"/>
      </sch:rule>
    </sch:pattern>
  </appinfo>
</annotation>
</element>
<!-- ===== -->
<complexType name="DirectedNodePropertyType">
<choice>
  <element ref="gml:Node" minOccurs="0"/>
</choice>
<attribute name="orientation" type="gml:SignType" default="+"/>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- primitive topology objects (1-dimensional) -->
<!-- ===== -->
<element name="Edge" type="gml:EdgeType" substitutionGroup="gml:_TopoPrimitive"/>
<!-- ===== -->
<complexType name="EdgeType">
<annotation>
  <documentation>There is precisely one positively directed and one negatively directed node in the boundary of every edge. The negatively and positively directed nodes correspond to the start and end nodes respectively. The optional coboundary of an edge is a circular sequence of directed faces which are incident on this edge in document order. Faces which use a particular boundary edge in its positive orientation appear with positive orientation on the coboundary of the same edge. In the 2D case, the orientation of the face on the left of the edge is "+"; the orientation of the face on the right on its right is "-". An edge may optionally be realised by a 1-dimensional (curve) geometric primitive.</documentation>
</annotation>
<complexContent>
  <extension base="gml:AbstractTopoPrimitiveType">
    <sequence>
      <element ref="gml:directedNode" minOccurs="2" maxOccurs="2"/>
      <element ref="gml:directedFace" minOccurs="0" maxOccurs="unbounded"/>
      <element ref="gml:curveProperty" minOccurs="0"/>
    </sequence>
  </extension>
</complexContent>
</complexType>
<!-- ===== Property for topology association - by Value or by Reference ===== -->
<element name="directedEdge" type="gml:DirectedEdgePropertyType">
<annotation>
  <appinfo>
    <sch:pattern>
      <sch:rule context="gml:directedEdge">
        <sch:extends rule="hrefOrContent"/>
      </sch:rule>
    </sch:pattern>
  </appinfo>
</annotation>
</element>
<!-- ===== -->
<complexType name="DirectedEdgePropertyType">
<choice>
  <element ref="gml:Edge" minOccurs="0"/>
</choice>
<attribute name="orientation" type="gml:SignType" default="+"/>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

```

```

<!-- ===== -->
<!-- primitive topology objects (2-dimensional) -->
<!-- ===== -->
<element name="Face" type="gml:FaceType" substitutionGroup="gml:_TopoPrimitive"/>
<!-- ===== -->
<complexType name="FaceType">
  <annotation>
    <documentation>
      The topological boundary of a face consists of a set of directed edges. Note that all edges associated with a Face, including dangling and interior edges, appear in the boundary. Dangling and interior edges are each referenced by pairs of directedEdges with opposing orientations. The optional coboundary of a face is a pair of directed solids which are bounded by this face. If present, there is precisely one positively directed and one negatively directed solid in the coboundary of every face. The positively directed solid corresponds to the solid which lies in the direction of the positively directed normal to the face in any geometric realisation. A face may optionally be realised by a 2-dimensional (surface) geometric primitive.</documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractTopoPrimitiveType">
        <sequence>
          <element ref="gml:directedEdge" maxOccurs="unbounded"/>
          <element ref="gml:directedTopoSolid" minOccurs="0" maxOccurs="2"/>
          <element ref="gml:surfaceProperty" minOccurs="0"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
<!-- ===== Property for topology association - by Value or by Reference ===== -->
<element name="directedFace" type="gml:DirectedFacePropertyType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:directedFace">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
  </annotation>
</element>
<!-- ===== -->
<complexType name="DirectedFacePropertyType">
  <choice>
    <element ref="gml:Face" minOccurs="0"/>
  </choice>
  <attribute name="orientation" type="gml:SignType" default="+"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- primitive topology objects (3-dimensional) -->
<!-- ===== -->
<element name="TopoSolid" type="gml:TopoSolidType" substitutionGroup="gml:_TopoPrimitive"/>
<!-- ===== -->
<complexType name="TopoSolidType">
  <annotation>
    <documentation>
      The topological boundary of a TopoSolid consists of a set of directed faces. Note that all faces associated with the TopoSolid, including dangling faces, appear in the boundary. The coboundary of a TopoSolid is empty and hence requires no representation.</documentation>
    </annotation>
    <complexContent>
      <extension base="gml:AbstractTopoPrimitiveType">
        <sequence>
          <element ref="gml:directedFace" maxOccurs="unbounded"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
<!-- ===== -->
<!-- ===== Property for topology association - by Value or by Reference ===== -->
<element name="directedTopoSolid" type="gml:DirectedTopoSolidPropertyType">
  <annotation>
    <appinfo>

```

```

<sch:pattern>
  <sch:rule context="gml:directedTopoSolid">
    <sch:extends rule="hrefOrContent"/>
  </sch:rule>
</sch:pattern>
<appinfo>
</annotation>
</element>
<!-- ===== -->
<complexType name="DirectedTopoSolidPropertyType">
  <choice>
    <element ref="gml:TopoSolid"/>
  </choice>
  <attribute name="orientation" type="gml:SignType" default="+"/>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<element name="TopoPoint" type="gml:TopoPointType"/>
<!-- ===== -->
<complexType name="TopoPointType">
  <annotation>
    <documentation>The intended use of TopoPoint is to appear within a point feature to express the structural and possibly geometric relationships of this point to other features via shared node definitions. Note the orientation assigned to the directedNode has no meaning in this context. It is preserved for symmetry with the types and elements which follow.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTopologyType">
      <sequence>
        <element ref="gml:directedNode"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== -->
<!-- ===== Property for topology association - by Value ===== -->
<element name="topoPointProperty" type="gml:TopoPointPropertyType"/>
<!-- ===== -->
<complexType name="TopoPointPropertyType">
  <sequence>
    <element ref="gml:TopoPoint"/>
  </sequence>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="TopoCurve" type="gml:TopoCurveType"/>
<!-- ===== -->
<complexType name="TopoCurveType">
  <annotation>
    <documentation> The end Node of each directedEdge of a TopoCurveType is the start Node of the next directedEdge of the TopoCurveType in document order. The TopoCurve type and element represent a homogeneous topological expression, a list of directed edges, which if realised are isomorphic to a geometric curve primitive. The intended use of TopoCurve is to appear within a line feature instance to express the structural and geometric relationships of this line to other features via the shared edge definitions.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractTopologyType">
      <sequence>
        <element ref="gml:directedEdge" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<!-- ===== Property for topology association - by Value ===== -->
<element name="topoCurveProperty" type="gml:TopoCurvePropertyType"/>
<!-- ===== -->
<complexType name="TopoCurvePropertyType">
  <sequence>
    <element ref="gml:TopoCurve"/>
  </sequence>
</complexType>

```

```

        </sequence>
    </complexType>
    <!-- ===== -->
    <!-- ===== -->
    <element name="TopoSurface" type="gml:TopoSurfaceType"/>
    <!-- ===== -->
    <complexType name="TopoSurfaceType">
        <annotation>
            <documentation>The TopoSurface type and element represent a homogeneous topological expression, a set of directed faces, which if realised are isomorphic to a geometric surface primitive. The intended use of TopoSurface is to appear within a surface feature instance to express the structural and possibly geometric relationships of this surface to other features via the shared face definitions.</documentation>
        </annotation>
        <complexContent>
            <extension base="gml:AbstractTopologyType">
                <sequence>
                    <element ref="gml:directedFace" maxOccurs="unbounded"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <!-- ===== -->
    <!-- ===== Property for topology association - by Value ===== -->
    <element name="topoSurfaceProperty" type="gml:TopoSurfacePropertyType"/>
    <!-- ===== -->
    <complexType name="TopoSurfacePropertyType">
        <sequence>
            <element ref="gml:TopoSurface"/>
        </sequence>
    </complexType>
    <!-- ===== -->
    <element name="TopoVolume" type="gml:TopoVolumeType"/>
    <!-- ===== -->
    <complexType name="TopoVolumeType">
        <annotation>
            <documentation>The TopoVolume type and element represent a homogeneous topological expression, a set of directed TopoSolids, which if realised are isomorphic to a geometric solid primitive. The intended use of TopoVolume is to appear within a 3D solid feature instance to express the structural and geometric relationships of this solid to other features via the shared TopoSolid definitions. . Note the orientation assigned to the directedSolid has no meaning in three dimensions. It is preserved for symmetry with the preceding types and elements.</documentation>
        </annotation>
        <complexContent>
            <extension base="gml:AbstractTopologyType">
                <sequence>
                    <element ref="gml:directedTopoSolid" maxOccurs="unbounded"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <!-- ===== -->
    <!-- ===== Property for topology association - by Value ===== -->
    <element name="topoVolumeProperty" type="gml:TopoVolumePropertyType"/>
    <!-- ===== -->
    <complexType name="TopoVolumePropertyType">
        <sequence>
            <element ref="gml:TopoVolume"/>
        </sequence>
    </complexType>
    <!-- ===== -->
    <!-- ===== -->
    <element name="TopoComplex" type="gml:TopoComplexType" substitutionGroup="gml:_Topology"/>
    <!-- ===== -->
    <complexType name="TopoComplexType">
        <annotation>
            <documentation>This type represents a TP_Complex capable of holding topological primitives.</documentation>
        </annotation>
        <complexContent>
            <extension base="gml:AbstractTopologyType">
                <sequence>

```

```

<element ref="gml:maximalComplex"/>
<element ref="gml:superComplex" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:subComplex" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:topoPrimitiveMember" minOccurs="0" maxOccurs="unbounded"/>
<element ref="gml:topoPrimitiveMembers" minOccurs="0"/>
</sequence>
<attribute name="isMaximal" type="boolean" default="false"/>
</extension>
</complexContent>
</complexType>
<!-- ===== Property for topology association - by Value or Reference ===== -->
<element name="topoComplexProperty" type="gml:TopoComplexMemberType"/>
<!-- ===== -->
<element name="subComplex" type="gml:TopoComplexMemberType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:subComplex">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
</annotation>
</element>
<!-- ===== -->
<element name="superComplex" type="gml:TopoComplexMemberType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:superComplex">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
</annotation>
</element>
<!-- ===== -->
<element name="maximalComplex" type="gml:TopoComplexMemberType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:subComplex">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
<documentation>Need schamatron test here that isMaximal attribute value is true</documentation>
</annotation>
</element>
<!-- ===== -->
<complexType name="TopoComplexMemberType">
<annotation>
<documentation>This Property can be used to embed a TopoComplex in a feature collection.</documentation>
</annotation>
<sequence>
<element ref="gml:TopoComplex" minOccurs="0"/>
</sequence>
<attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- ===== Property for topology association - by Value or Reference ===== -->
<element name="topoPrimitiveMember" type="gml:topoPrimitiveMemberType">
<annotation>
<appinfo>
<sch:pattern>
<sch:rule context="gml:topoPrimitiveMember">
<sch:extends rule="hrefOrContent"/>
</sch:rule>
</sch:pattern>
</appinfo>
</annotation>

```

```

</element>
<!-- ===== -->
<complexType name="topoPrimitiveMemberType">
  <annotation>
    <documentation>This type supports embedding topological primitives in a TopoComplex.</documentation>
  </annotation>
  <sequence>
    <element ref="gml:_TopoPrimitive" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- ===== Property for topology association - by Value ===== -->
<element name="topoPrimitiveMembers" type="gml:TopoPrimitiveArrayAssociationType">
  <annotation>
    <appinfo>
      <sch:pattern>
        <sch:rule context="gml:topoPrimitiveMember">
          <sch:extends rule="hrefOrContent"/>
        </sch:rule>
      </sch:pattern>
    </appinfo>
  </annotation>
</element>
<!-- ===== -->
<complexType name="TopoPrimitiveArrayAssociationType">
  <annotation>
    <documentation>This type supports embedding an array of topological primitives in a TopoComplex</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:ArrayAssociationType">
      <sequence>
        <element ref="gml:_TopoPrimitive" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
</schema>

```

## C.28 units.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" version="3.00" xml:lang="en">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-units:v3.00"/>
    <documentation>
      <name>units.xsd</name>
      <version>3.0</version>
      <scope>How to encode units of measure (or uom) for numerical values. </scope>
      <description>Builds on gmlBase.xsd to encode units of measure (or uom), including definitions of units of measure and dictionaries of such definitions. GML 3.0 candidate schema, primary editor: Arliss Whiteside. Last updated 2002/11/13.</description>
      <copyright>Copyright (c) 2001-2002 OpenGIS, All Rights Reserved.</copyright>
      <conformance>Parts of this schema are based on Subclause 6.5.7 of ISO/CD 19103 Geographic information - Conceptual schema language, on Subclause A.5.2.2.3 of ISO/CD 19118 Geographic information - Encoding, and on most of OpenGIS Recommendation Paper OGC 02-007r4 Units of Measure Use and Definition Recommendations. </conformance>
    </documentation>
  </annotation>
  <!-- ===== -->
  includes and imports
  =====-->
  <include schemaLocation="dictionary.xsd"/>
  <!-- ===== -->
  elements and types
  =====-->

```

```

<element name="unitOfMeasure" type="gml:UnitOfMeasureType"/>
<!-- ===== -->
<complexType name="UnitOfMeasureType">
  <annotation>
    <documentation>Reference to a unit of measure definition that applies to all the numerical values described by the element containing this element. Notice that a complexType which needs to include the uom attribute can do so by extending this complexType. Alternately, this complexType can be used as a pattern for a new complexType. </documentation>
  </annotation>
  <sequence>
    <attribute name="uom" type="anyURI!" use="required">
      <annotation>
        <documentation>Reference to a unit of measure definition, usually within the same XML document but possibly outside the XML document which contains this reference. For a reference within the same XML document, the "#" symbol should be used, followed by a text abbreviation of the unit name. However, the "#" symbol may be optional, and still may be interpreted as a reference. </documentation>
      </annotation>
    </attribute>
  </complexType>
<!-- ===== -->
<element name="UnitDefinition" type="gml:UnitDefinitionType" substitutionGroup="gml:Definition"/>
<!-- ===== -->
<complexType name="UnitDefinitionType">
  <annotation>
    <documentation>Definition of a unit of measure (or uom). The definition includes a quantityType property, which indicates the phenomenon to which the units apply, and a catalogSymbol, which gives the short symbol used for this unit. This element is used when the relationship of this unit to other units or units systems is unknown.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:DefinitionType">
      <sequence>
        <element ref="gml:quantityType"/>
        <element ref="gml:catalogSymbol" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="BaseUnit" type="gml:BaseUnitType" substitutionGroup="gml:UnitDefinition"/>
<!-- ===== -->
<complexType name="BaseUnitType">
  <annotation>
    <documentation>Definition of a unit of measure which is a base unit from the system of units. A base unit cannot be derived by combination of other base units within this system. Sometimes known as "fundamental unit". </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:UnitDefinitionType">
      <sequence>
        <element name="unitsSystem" type="gml:ReferenceType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="DerivedUnit" type="gml:DerivedUnitType" substitutionGroup="gml:UnitDefinition"/>
<!-- ===== -->
<complexType name="DerivedUnitType">
  <annotation>
    <documentation>Definition of a unit of measure which is defined through algebraic combination of more primitive units, which are usually base units from a particular system of units. Derived units based directly on base units are usually preferred for quantities other than the base units or fundamental quantities within a system. If a derived unit is not the preferred unit, the ConventionalUnit element should be used instead.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:UnitDefinitionType">
      <sequence>
        <element ref="gml:derivationUnitTerm" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->

```

```

<element name="ConventionalUnit" type="gml:ConventionalUnitType" substitutionGroup="gml:UnitDefinition"/>
<!-- ===== -->
<complexType name="ConventionalUnitType">
  <annotation>
    <documentation>Definition of a unit of measure which is related to a preferred unit for this quantity type through a conversion formula. A method for deriving this unit by algebraic combination of more primitive units, may also be provided.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:UnitDefinitionType">
      <sequence>
        <choice>
          <element ref="gml:conversionToPreferredUnit"/>
          <element ref="gml:roughConversionToPreferredUnit"/>
        </choice>
        <element ref="gml:derivationUnitTerm" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="quantityType" type="gml:StringOrRefType">
  <annotation>
    <documentation>Informal description of the phenomenon or type of quantity that is measured or observed. For example, "length", "angle", "time", "pressure", or "temperature". When the quantity is the result of an observation or measurement, this term is known as Observable Type or Measurand. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="catalogSymbol" type="gml:CodeType">
  <annotation>
    <documentation>For global understanding of a unit of measure, it is often possible to reference an item in a catalog of units, using a symbol in that catalog. The "codeSpace" attribute in "CodeType" identifies a namespace for the catalog symbol value, and might reference the catalog. The "string" value in "CodeType" contains the value of a symbol that is unique within this catalog namespace. This symbol often appears explicitly in the catalog, but it could be a combination of symbols using a specified algebra of units. For example, the symbol "cm" might indicate that it is the "m" symbol combined with the "c" prefix.</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="derivationUnitTerm" type="gml:DerivationUnitTermType">
<!-- ===== -->
<complexType name="DerivationUnitTermType">
  <annotation>
    <documentation>Definition of one unit term for a derived unit of measure. This unit term references another unit of measure (uom) and provides an integer exponent applied to that unit in defining the compound unit. The exponent can be positive or negative, but not zero. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:UnitOfMeasureType">
      <attribute name="exponent" type="integer"/>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<element name="conversionToPreferredUnit" type="gml:ConversionToPreferredUnitType">
  <annotation>
    <documentation>This element is included when this unit has an accurate conversion to the preferred unit for this quantity type. </documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="roughConversionToPreferredUnit" type="gml:ConversionToPreferredUnitType">
  <annotation>
    <documentation>This element is included when the correct definition of this unit is unknown, but this unit has a rough or inaccurate conversion to the preferred unit for this quantity type. </documentation>
  </annotation>
</element>
<!-- ===== -->

```

```

<complexType name="ConversionToPreferredUnitType">
  <annotation>
    <documentation>Relation of a unit to the preferred unit for this quantity type, specified by an arithmetic conversion (scaling and/or offset). A preferred unit is either a base unit or a derived unit selected for all units of one quantity type. The mandatory attribute "uom" shall reference the preferred unit that this conversion applies to. The conversion is specified by one of two alternative elements: "factor" or "formula". </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:UnitOfMeasureType">
      <choice>
        <element name="factor" type="double">
          <annotation>
            <documentation>Specification of the scale factor by which a value using this unit of measure can be multiplied to obtain the corresponding value using the preferred unit of measure. </documentation>
          </annotation>
        </element>
        <element name="formula" type="gml:FormulaType">
          <annotation>
            <documentation>Specification of the formula by which a value using this unit of measure can be converted to obtain the corresponding value using the preferred unit of measure. </documentation>
          </annotation>
        </element>
      </choice>
    </extension>
  </complexContent>
</complexType>
<!-- ===== -->
<complexType name="FormulaType">
  <annotation>
    <documentation>Parameters of a simple formula by which a value using this unit of measure can be converted to the corresponding value using the preferred unit of measure. The formula element contains elements a, b, c and d, whose values use the XML Schema type "double". These values are used in the formula  $y = (a + bx) / (c + dx)$ , where x is a value using this unit, and y is the corresponding value using the preferred unit. The elements a and d are optional, and if values are not provided, those parameters are considered to be zero. If values are not provided for both a and d, the formula is equivalent to a fraction with numerator and denominator parameters. </documentation>
  </annotation>
  <sequence>
    <element name="a" type="double" minOccurs="0"/>
    <element name="b" type="double"/>
    <element name="c" type="double"/>
    <element name="d" type="double" minOccurs="0"/>
  </sequence>
</complexType>
<!-- ===== -->
</schema>

```

## C.29 valueObjects.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:sch="http://www.ascc.net/xml/schematron" xmlns:gml="http://www.opengis.net/gml"
  xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDefault="qualified" attributeFormDefault="unqualified" version="3.1.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xsd:valueObjects:v3.1.0">valueObjects.xsd</appinfo>
    <documentation> Copyright (c) 2002 OGC, All Rights Reserved. GML conformant schema for Values in which the * scalar Value types and lists have their values recorded in simpleContent elements * complex Value types are built recursively
  </documentation>
  </annotation>
  <!-- ===== -->
  <!-- geometry and temporal included so that _Geometry and _TimeObject can be added to Value choice group -->
  <include schemaLocation="geometryBasic0d1d.xsd"/>
  <include schemaLocation="temporal.xsd"/>
  <!-- ===== -->
  <group name="Value">
    <choice>
      <element ref="gml:_Value"/>
      <element ref="gml:_Geometry"/>
      <element ref="gml:_TimeObject"/>
      <element ref="gml:Null"/>
      <element ref="gml:measure"/>
    </choice>
  </group>

```

```

        </choice>
        <!-- <xs:documentation> Utility choice group which unifies generic Values defined in this schema
document with Geometry and Temporal objects and the Measures described above, so that any of these may be used within
aggregate Values. </xs:annotation> </xs:documentation> -->
</group>
<!-- ===== -->
<element name="_Value" abstract="true" substitutionGroup="gml:_Object">
    <annotation>
        <documentation>Abstract element which acts as the head of a substitution group which contains _ScalarValue,
_ValueList and CompositeValue and (transitively) the elements in their substitution groups. This element may be used in
an application schema as a variable, so that in an XML instance document any member of its substitution group may occur.
</documentation>
    </annotation>
</element>
<!-- ===== -->
<!-- ===== Scalar Values ===== -->
<element name="_ScalarValue" abstract="true" substitutionGroup="gml:_Value">
    <annotation>
        <documentation>Abstract element which acts as the head of a substitution group which contains Boolean, Category,
Count and Quantity, and (transitively) the elements in their substitution groups. This element may be used in an application
schema as a variable, so that in an XML instance document any member of its substitution group may occur. </documentation>
    </annotation>
</element>
<!-- ===== -->
<element name="_ScalarValueList" abstract="true" substitutionGroup="gml:_Value">
    <annotation>
        <documentation>Abstract element which acts as the head of a substitution group which contains the compact
encodings BooleanList, CategoryList, CountList and QuantityList, and (transitively) the elements in their substitution groups. This
element may be used in an application schema as a variable, so that in an XML instance document any member of its
substitution group may occur. </documentation>
    </annotation>
</element>
<!-- ===== -->
<!-- ===== Boolean ===== -->
<element name="Boolean" type="boolean" substitutionGroup="gml:_ScalarValue">
    <annotation>
        <documentation>A value from two-valued logic, using the XML Schema boolean type. An instance may take the
values {true, false, 1, 0}. </documentation>
    </annotation>
</element>
<element name="BooleanList" type="gml:booleanOrNullList" substitutionGroup="gml:_ScalarValueList">
    <annotation>
        <documentation>XML List based on XML Schema boolean type. An element of this type contains a space-separated
list of boolean values {0,1,true,false}</documentation>
    </annotation>
</element>
<!-- ===== -->
<!-- ===== Category ===== -->
<element name="Category" type="gml:CodeType" substitutionGroup="gml:_ScalarValue">
    <annotation>
        <documentation>A term representing a classification. It has an optional XML attribute codeSpace, whose value is a
URI which identifies a dictionary, codelist or authority for the term. </documentation>
    </annotation>
</element>
<element name="CategoryList" type="gml:CodeOrNullListType" substitutionGroup="gml:_ScalarValueList">
    <annotation>
        <documentation>A space-separated list of terms or nulls. A single XML attribute codeSpace may be provided, which
authorises all the terms in the list. </documentation>
    </annotation>
</element>
<!-- ===== -->
<!-- ===== Quantity ===== -->
<element name="Quantity" type="gml:MeasureType" substitutionGroup="gml:_ScalarValue">
    <annotation>
        <documentation>A numeric value with a scale. The content of the element is an amount using the XML Schema type
double which permits decimal or scientific notation. An XML attribute uom ("unit of measure") is required, whose value is a URI
which identifies the definition of the scale or units by which the numeric value must be multiplied. </documentation>
    </annotation>
</element>

```

```

<element name="QuantityList" type="gml:MeasureOrNullListType" substitutionGroup="gml:_ScalarValueList">
  <annotation>
    <documentation>A space separated list of amounts or nulls. The amounts use the XML Schema type double. A single XML attribute uom ("unit of measure") is required, whose value is a URI which identifies the definition of the scale or units by which all the amounts in the list must be multiplied. </documentation>
  </annotation>
</element>
<!-- ===== Count ===== -->
<element name="Count" type="integer" substitutionGroup="gml:_ScalarValue">
  <annotation>
    <documentation>An integer representing a frequency of occurrence. </documentation>
  </annotation>
</element>
<element name="CountList" type="gml:integerOrNullList" substitutionGroup="gml:_ScalarValueList">
  <annotation>
    <documentation>A space-separated list of integers or nulls. </documentation>
  </annotation>
</element>
<!-- ===== aggregate Value types ===== -->
<!-- ===== ValueCollection ===== -->
<complexType name="CompositeValueType">
  <annotation>
    <documentation>Aggregate value built from other Values using the Composite pattern. It contains zero or an arbitrary number of valueComponent elements, and zero or one valueComponents elements. It may be used for strongly coupled aggregates (vectors, tensors) or for arbitrary collections of values.</documentation>
  </annotation>
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <sequence>
        <element ref="gml:valueComponent" minOccurs="0" maxOccurs="unbounded"/>
        <element ref="gml:valueComponents" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<element name="CompositeValue" type="gml:CompositeValueType" substitutionGroup="gml:_Value">
  <annotation>
    <documentation>Aggregate value built using the Composite pattern. </documentation>
  </annotation>
</element>
<!-- ===== ValueArray ===== -->
<complexType name="ValueArrayType">
  <annotation>
    <documentation>A Value Array is used for homogeneous arrays of primitive and aggregate values. The member values may be scalars, composites, arrays or lists. ValueArray has the same content model as CompositeValue, but the member values must be homogeneous. The element declaration contains a Schematron constraint which expresses this restriction precisely. Since the members are homogeneous, the referenceSystem (uom, codeSpace) may be specified on the ValueArray itself and implicitly inherited by all the members if desired. Note that a_ScalarValueList is preferred for arrays of Scalar Values since this is a more efficient encoding. </documentation>
  </annotation>
  <complexContent>
    <extension base="gml:CompositeValueType">
      <attributeGroup ref="gml:referenceSystem"/>
    </extension>
  </complexContent>
</complexType>
<element name="ValueArray" type="gml:ValueArrayType" substitutionGroup="gml:CompositeValue">
  <annotation>
    <appinfo>
      <sch:pattern name="Check either codeSpace or uom not both">
        <sch:rule context="gml:ValueArray">
          <sch:report test="@codeSpace and @uom">ValueArray may not carry both a reference to a codeSpace and a uom</sch:report>
        </sch:rule>
      </sch:pattern>
      <sch:pattern name="Check components are homogeneous">
        <sch:rule context="gml:ValueArray">

```

and a uom</sch:report>

```

        <sch:assert test="count(gml:valueComponent/*) = count(gml:valueComponent/*[name() = name(..../gml:valueComponent[1]/*[1]))">All components of <sch:name/> must be of the same type</sch:assert>
        <sch:assert test="count(gml:valueComponents/*) = count(gml:valueComponents/*[name() = name(../*[1]))">All components of <sch:name/> must be of the same type</sch:assert>
    </sch:rule>
</sch:pattern>
</appinfo>
<documentation>A Value Array is used for homogeneous arrays of primitive and aggregate values.
 is preferred for arrays of Scalar Values since this is more efficient. Since "choice" is not available for attribute groups, an external constraint (e.g. Schematron) would be required to enforce the selection of only one of these through schema validation </documentation>
</annotation>
</element>
<!-- attribute group required for ValueArray -->
<attributeGroup name="referenceSystem">
    <attribute name="codeSpace" type="anyURI" use="optional"/>
    <attribute name=" uom" type="anyURI" use="optional"/>
</attributeGroup>
<!-- =====-->
<!-- =====-->
<!-- =====-->
<element name="QuantityExtent" type="gml:QuantityExtentType" substitutionGroup="gml:_Value">
    <annotation>
        <documentation>Utility element to store a 2-point range of numeric values. If one member is a null, then this is a single ended interval. </documentation>
    </annotation>
</element>
<!-- -->
<complexType name="QuantityExtentType">
    <annotation>
        <documentation>Restriction of list type to store a 2-point range of numeric values. If one member is a null, then this is a single ended interval. </documentation>
    </annotation>
    <simpleContent>
        <restriction base="gml:MeasureOrNullListType">
            <length value="2"/>
        </restriction>
    </simpleContent>
</complexType>
<!-- =====-->
<element name="CategoryExtent" type="gml:CategoryExtentType" substitutionGroup="gml:_Value">
    <annotation>
        <documentation>Utility element to store a 2-point range of ordinal values. If one member is a null, then this is a single ended interval. </documentation>
    </annotation>
</element>
<!-- -->
<complexType name="CategoryExtentType">
    <annotation>
        <documentation>Restriction of list type to store a 2-point range of ordinal values. If one member is a null, then this is a single ended interval. </documentation>
    </annotation>
    <simpleContent>
        <restriction base="gml:CodeOrNullListType">
            <length value="2"/>
        </restriction>
    </simpleContent>
</complexType>
<!-- =====-->
<element name="CountExtent" type="gml:CountExtentType" substitutionGroup="gml:_Value">
    <annotation>
        <documentation>Utility element to store a 2-point range of frequency values. If one member is a null, then this is a single ended interval. </documentation>
    </annotation>
</element>
<!-- -->
<simpleType name="CountExtentType">
    <annotation>
        <documentation>Restriction of list type to store a 2-point range of frequency values. If one member is a null, then this is a single ended interval. </documentation>
    </annotation>

```

```
</annotation>
<restriction base="gml:integerOrNullList">
  <length value="2"/>
</restriction>
</simpleType>
<!-- ===== pieces needed for compositing -->
<element name="valueProperty" type="gml:Value.PropertyType">
  <annotation>
    <documentation>Element which refers to, or contains, a Value</documentation>
  </annotation>
</element>
<!-- ===== -->
<element name="valueComponent" type="gml:Value.PropertyType">
  <annotation>
    <documentation>Element which refers to, or contains, a Value. This version is used in CompositeValues.</documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="Value.PropertyType">
  <annotation>
    <documentation>GML property which refers to, or contains, a Value</documentation>
  </annotation>
  <sequence>
    <group ref="gml:Value" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<!-- ===== -->
<element name="valueComponents" type="gml:Value.Array.PropertyType">
  <annotation>
    <documentation>Element which refers to, or contains, a set of homogeneously typed Values. </documentation>
  </annotation>
</element>
<!-- ===== -->
<complexType name="Value.Array.PropertyType">
  <annotation>
    <documentation>GML property which refers to, or contains, a set of homogeneously typed Values. </documentation>
  </annotation>
  <sequence>
    <group ref="gml:Value" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<!-- ===== utility typed valueProperty types -->
<complexType name="ScalarValue.PropertyType">
  <annotation>
    <documentation>Property whose content is a scalar value.</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:Value.PropertyType">
      <sequence>
        <element ref="gml:_ScalarValue" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<complexType name="Boolean.PropertyType">
  <annotation>
    <documentation>Property whose content is a Boolean value.</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:Value.PropertyType">
      <sequence>
        <element ref="gml:Boolean" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<complexType name="Category.PropertyType">
```

```

<annotation>
  <documentation>Property whose content is a Category.</documentation>
</annotation>
<complexContent>
  <restriction base="gml:ValuePropertyType">
    <sequence>
      <element ref="gml:Category" minOccurs="0"/>
    </sequence>
  </restriction>
</complexContent>
</complexType>
<complexType name="Quantity.PropertyType">
  <annotation>
    <documentation>Property whose content is a Quantity.</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:ValuePropertyType">
      <sequence>
        <element ref="gml:Quantity" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<complexType name="Count.PropertyType">
  <annotation>
    <documentation>Property whose content is a Count.</documentation>
  </annotation>
  <complexContent>
    <restriction base="gml:ValuePropertyType">
      <sequence>
        <element ref="gml:Count" minOccurs="0"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
<!-- ===== -->
</schema>

```

### C.30 xlink.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- File: xlink.xsd -->
<schema targetNamespace="http://www.w3.org/1999/xlink" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
       xmlns="http://www.w3.org/2001/XMLSchema" xmlns:xlink="http://www.w3.org/1999/xlink" version="2.0">
  <annotation>
    <appinfo source="urn:opengis:specification:gml:schema-xlinks:v3.0c2">xlinks.xsd v3.0b2 2001-07</appinfo>
    <documentation>
      GML 3.0 candidate xlink schema. Copyright (c) 2001 OGC, All Rights Reserved.
    </documentation>
  </annotation>
  <!-- ===== -->
  global declarations
  <!-- ===== -->
  <!-- locator attribute -->
  <attribute name="href" type="anyURI"/>
  <!-- semantic attributes -->
  <attribute name="role" type="anyURI"/>
  <attribute name="arcrole" type="anyURI"/>
  <attribute name="title" type="string"/>
  <!-- behavior attributes -->
  <attribute name="show">
    <annotation>
      <documentation>
        The 'show' attribute is used to communicate the desired presentation
        of the ending resource on traversal from the starting resource; it's
        value should be treated as follows:
        new - load ending resource in a new window, frame, pane, or other
        presentation context
      </documentation>
    </annotation>
  </attribute>
</schema>

```

replace - load the resource in the same window, frame, pane, or other presentation context  
 embed - load ending resource in place of the presentation of the starting resource  
 other - behavior is unconstrained; examine other markup in the link for hints  
 none - behavior is unconstrained

```
</documentation>
</annotation>
<simpleType>
  <restriction base="string">
    <enumeration value="new"/>
    <enumeration value="replace"/>
    <enumeration value="embed"/>
    <enumeration value="other"/>
    <enumeration value="none"/>
  </restriction>
</simpleType>
</attribute>
<attribute name="actuate">
  <annotation>
    <documentation>
```

The 'actuate' attribute is used to communicate the desired timing of traversal from the starting resource to the ending resource; its value should be treated as follows:

onLoad - traverse to the ending resource immediately on loading the starting resource  
 onRequest - traverse from the starting resource to the ending resource only on a post-loading event triggered for this purpose  
 other - behavior is unconstrained; examine other markup in link for hints  
 none - behavior is unconstrained

```
</documentation>
</annotation>
<simpleType>
  <restriction base="string">
    <enumeration value="onLoad"/>
    <enumeration value="onRequest"/>
    <enumeration value="other"/>
    <enumeration value="none"/>
  </restriction>
</simpleType>
</attribute>
<!-- traversal attributes -->
<attribute name="label" type="string"/>
<attribute name="from" type="string"/>
<attribute name="to" type="string"/>
<!-- ===== Attributes grouped by XLink type, as specified in the W3C Proposed Recommendation (dated 2000-12-20) ===== -->
```

```
<attributeGroup name="simpleLink">
  <attribute name="type" type="string" fixed="simple" form="qualified"/>
  <attribute ref="xlink:href" use="optional"/>
  <attribute ref="xlink:role" use="optional"/>
  <attribute ref="xlink:arcrole" use="optional"/>
  <attribute ref="xlink:title" use="optional"/>
  <attribute ref="xlink:show" use="optional"/>
  <attribute ref="xlink:actuate" use="optional"/>
</attributeGroup>
<attributeGroup name="extendedLink">
  <attribute name="type" type="string" fixed="extended" form="qualified"/>
  <attribute ref="xlink:role" use="optional"/>
  <attribute ref="xlink:title" use="optional"/>
</attributeGroup>
<attributeGroup name="locatorLink">
  <attribute name="type" type="string" fixed="locator" form="qualified"/>
  <attribute ref="xlink:href" use="required"/>
  <attribute ref="xlink:role" use="optional"/>
  <attribute ref="xlink:title" use="optional"/>
```

```
<attribute ref="xlink:label" use="optional"/>
</attributeGroup>
<attributeGroup name="arclink">
  <attribute name="type" type="string" fixed="arc" form="qualified"/>
  <attribute ref="xlink:arcrole" use="optional"/>
  <attribute ref="xlink:title" use="optional"/>
  <attribute ref="xlink:show" use="optional"/>
  <attribute ref="xlink:actuate" use="optional"/>
  <attribute ref="xlink:from" use="optional"/>
  <attribute ref="xlink:to" use="optional"/>
</attributeGroup>
<attributeGroup name="resourceLink">
  <attribute name="type" type="string" fixed="resource" form="qualified"/>
  <attribute ref="xlink:role" use="optional"/>
  <attribute ref="xlink:title" use="optional"/>
  <attribute ref="xlink:label" use="optional"/>
</attributeGroup>
<attributeGroup name="titleLink">
  <attribute name="type" type="string" fixed="title" form="qualified"/>
</attributeGroup>
<attributeGroup name="emptyLink">
  <attribute name="type" type="string" fixed="none" form="qualified"/>
</attributeGroup>
</schema>
```

## Annex D (informative)

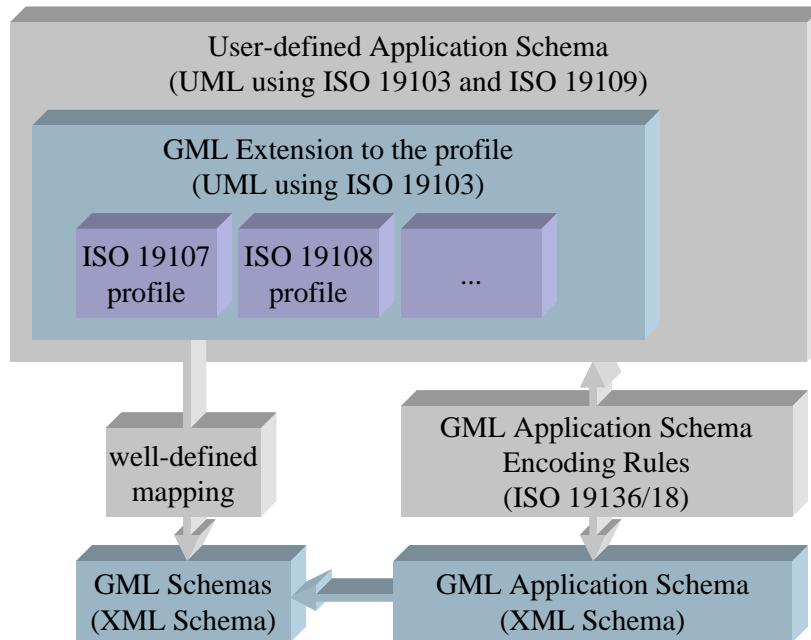
### ISO 19100 Profile and Application Schema of GML

#### D.1 General Approach

The approach taken by this standard is shown in the following diagram. The two main aspects are:

- Clear documentation of the conceptual model of GML: The profile of the ISO 19100 harmonized model that is implemented by GML is documented as well as the extensions to this profile.
- Support for application schema development either in UML or XML Schema: In order to achieve this two-way mapping between UML (i.e. ISO 19109 Application Schemas in UML) and XML Schema (i.e. GML Application Schemas in XML Schema) the constructs used in both representations have to be limited. While this reduces the expressiveness of the schema descriptions to some extent, this also reduces their complexity and may make it easier to implement them.

**NOTE** While the mapping from UML to XML Schema is discussed in Annex A of ISO DIS 19118, the reverse mapping is not discussed in any other ISO 19100 standard.



**Figure 12**

#### D.2 Profile of the ISO 19100 Harmonized Model used by GML

This section describes the profile of the ISO 19100 Harmonized Model that is used by GML. The description is based on version 4.5 of the model. In the description of the class diagrams of the profile the relationship and mapping to the GML Schemas is discussed.

NOTE Two types of differences between the ISO 19100 harmonized model and GML have been identified in this process:

- Differences in concepts. These shall be discussed, whether the difference is acceptable, GML is more appropriate or ISO 19100 is more appropriate.
- In general the encoding rules discussed in Annex E are used also in the encoding of the GML Schemas. However, since the GML Schemas were mostly handcrafted they exploit more of the specific capabilities of the implementation environment, i.e. XML and XML Schema. Examples are a number of predefined basic types (simple or complex types with simple content) or the use of global elements also for properties that shall be made substitutable (e.g. to define aliases for deprecated property names). These cases are documented in the following clauses or are straightforward.

Only elements from packages discussed below are part of the profile. All other elements are not part of the profile<sup>2</sup>.

Due to the nature of GML no operation of any class is part of the profile.

In addition, interface classes (stereotype <<Interface>>) without data structures and “Realization” relationships to classes without data structures have been deleted.

Furthermore, the navigability of associations has been restricted to the directions in which GML represents explicit object properties (most associations in the GML Schemas are navigable only in a single direction).

NOTE 1 All deprecated types, elements and attributes of GML are not considered in this annex.

NOTE 2 The general rules for the UML-to-XML-Schema mapping for GML Application Schemas is defined in Annex E.

NOTE 3 In this annex the namespace “xsd:” is used to refer to the namespace of XML Schema, which is “<http://www.w3.org/2001/XMLSchema>”. The namespace “gml:” refers to the namespace of GML, which is “<http://www.opengis.net/gml>”.

### D.2.1 ISO 19103 Conceptual Schema Language

In this section we define the basic types defined in ISO PDTS 19103 that are directly available in GML. In many cases simple types defined by XML Schema are used directly.

- “CharacterString” is implemented by xsd:string. The character encoding is defined in the processing instruction of the XML document (the default for XML documents is UTF-8).
- „Date“ is implemented by xsd:date. DatePrecision is not supported.
- “DateTime” is implemented by xsd:dateTime. Again, DatePrecision is not supported.
- “Time” is implemented by xsd:time.
- “Real” is implemented by xsd:double.
- “Decimal” is in general implemented by xsd:decimal. For practical reasons, often decimal values will also be represented in schemas by xsd:double.
- The generic basic type “Number” is in general implemented in GML Schemas by xsd:double.
- “Integer” is implemented by xsd:integer.

---

<sup>2</sup> The deleted packages are: ISO 19110, ISO 19112, ISO 19116, ISO 19117, ISO 19118, ISO 19119, ISO 19128, ISO 19133, ISO 19134, ISO GDF, ISO 00639 Human Language, all informative packages, New.

- “Boolean” is implemented by xsd:boolean.
- “Measure” is implemented by the simple type gml:MeasureType. The value is of type xsd:double, the uom-specifier is implemented by a URI to a <gml:UnitDefinition> element. The following subtypes are implemented by GML, each pointing to a corresponding unit definition (measure.xsd):
  - “Length” → gml:LengthType
  - “Scale” → gml:ScaleType
  - “Area” → gml:AreaType
  - “Volume” → gml:VolumeType
  - “Velocity” → gml:SpeedType

NOTE This terminology is incorrect in ISO 19103: velocity is a vector quantity, so may only be described by a single component when operating within a 1-D coordinate system.

- “Time” → gml:TimeType
- “Angle” → gml:AngleType
- “Vector” is implemented by gml:VectorType.
- “GenericName”, „LocalName“ and „ScopedName“ are implemented by gml:CodeType where the name space designator is a URI.
- “Measure” is implemented by gml:MeasureType, the uom-attribute is a URI pointing to the unit definition.
- “UnitOfMeasure” is implemented by gml:UnitDefinitionType.

ISO PDTS 19103 specifies that all NULL values are equivalent. GML uses a more explicit approach by requiring additional information about the reason for the NULL value. Whether an application uses this added information or not is optional.

## D.2.2 ISO 19107 Spatial Schema (Geometry)

### D.2.2.1 Overview

The UML model of the GML profile defined in this annex describes a conceptual model of the abstract types defined in ISO 19107. The same names for the classes and their properties as in ISO 19107 are used to document the GML profile to ease the comparison with that standard.

NOTE See chapter 2 of ISO 19107 for more details.

The following additional changes have been applied to the geometry package of ISO 19107.

**Table 10**

Change	Explanation
GM_Primitive: association „Interior to“ deleted	Currently not supported by GML
GM_Polygon: attribute „SpanningSurface“ deleted	Currently not supported by GML
GM_Solid: converted the operation “boundary()” to an attribute	As the boundary of GM_Solid is accessible only via the „boundary()“ operation, an attribute of the same name has been added. The attribute value is the result of the “boundary()” operation as defined in ISO 19107.

GM_Complex: association „Contains“ deleted	Currently not supported by GML
Derived attributes deleted in GM_MultiPrimitive subtypes	These attributes can be derived from the digital representation of the objects, therefore the redundant information has been omitted
GM_CompositePoint: deleted	GM_CompositePoint does not add any additional information. The type has been added in ISO 19107 for completeness only, but it is not expected that it will be used in GML documents. Therefore, it has been omitted in GML.
GM_PolynomialSpline has been made abstract	Currently not instantiable in GML, but the subtype GM_CubicSpline is.
GM_LineSegment: deleted	Not supported by GML, a GM_LineString with two control points shall be used instead
GM_CurveBoundary: deleted	Only used in operations
GM_ComplexBoundary: deleted	Only used in operations

NOTE GM\_OrientableCurve and GM\_OrientableSurface have been changed to “not abstract” (in accordance with ISO DIS 19107).

#### D.2.2.2 Geometry root

The following UML class diagrams illustrate the profile of the “Geometry root” package (compare with figures 5 to 6 of ISO 19107).

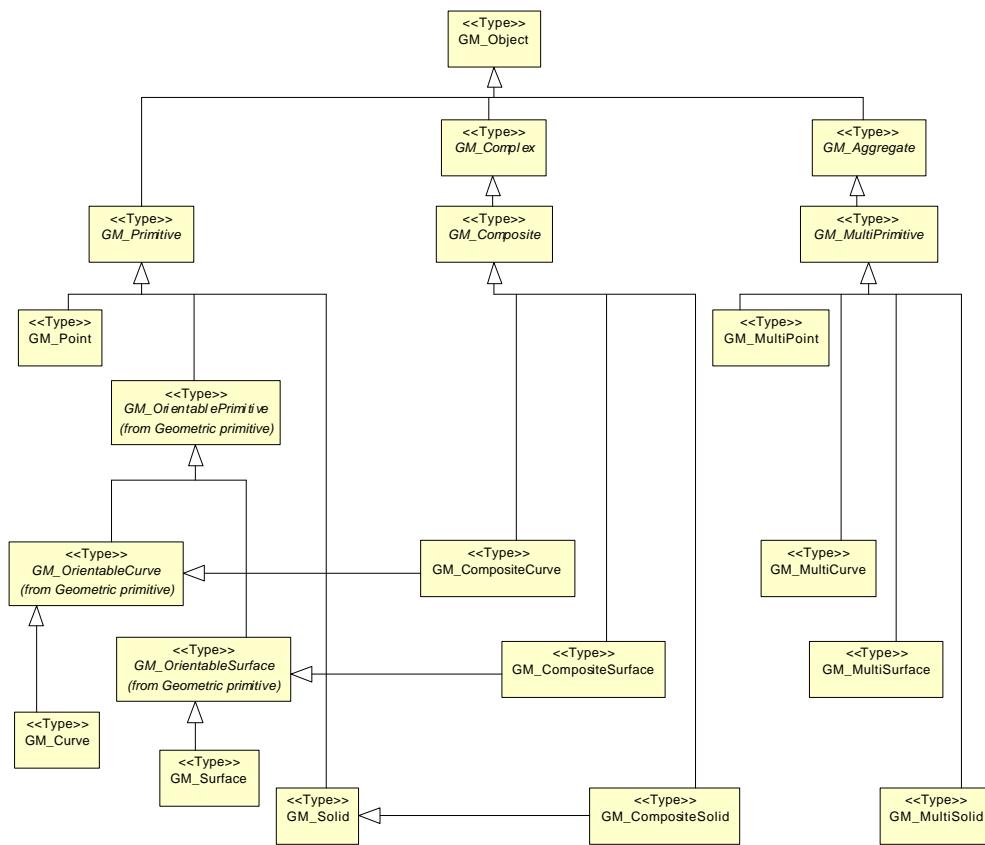
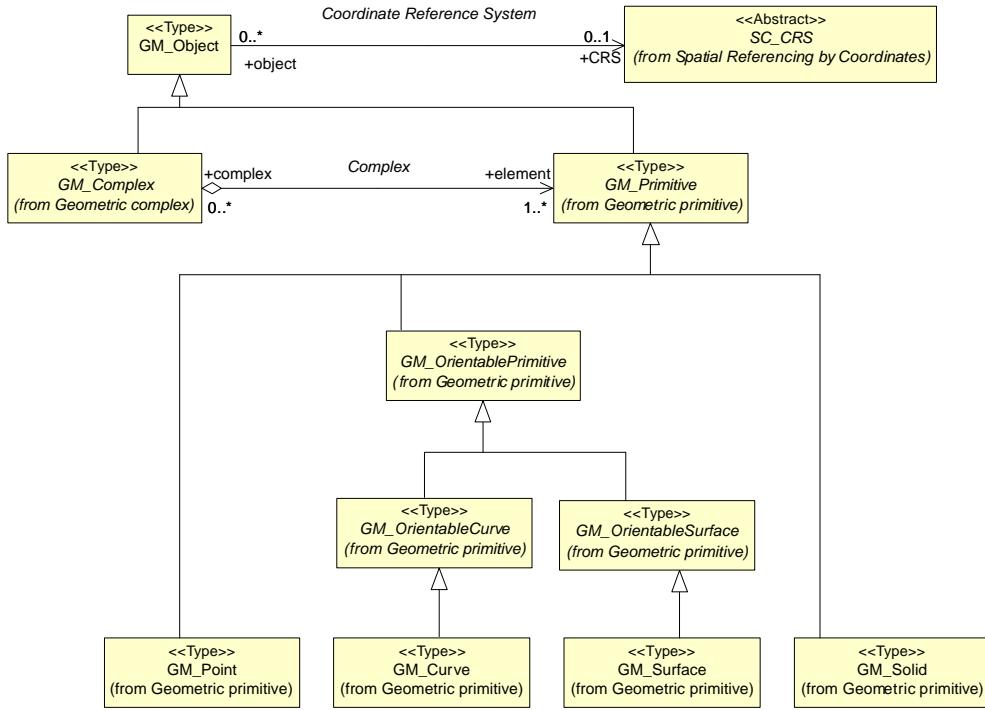


Figure 13



**Figure 14**

The mapping of the different classes to the GML Schemas is explained in the subsequent sections showing details of the class hierarchy.

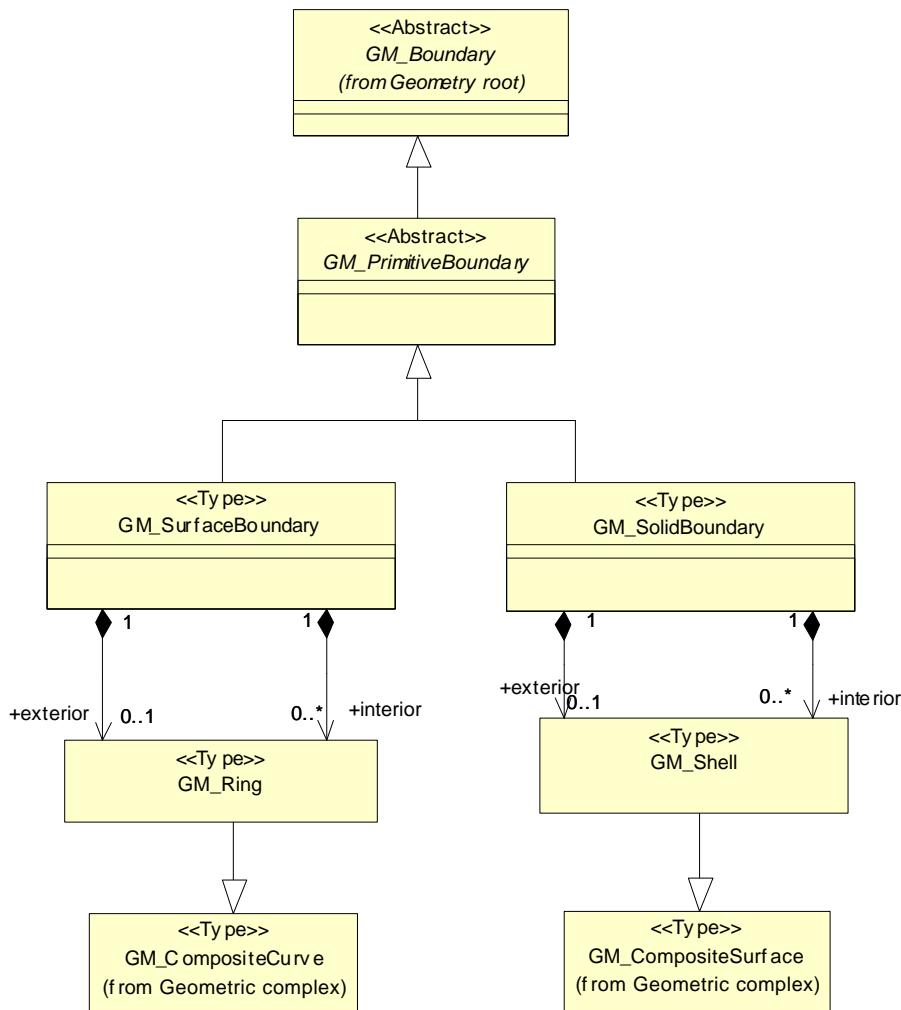
“GM\_Object” is represented by the “\_Geometry” object element, the “CRS” role is represented by the “srsName” property.

Differences are:

- The “\_Geometry” element can carry additional properties: an optional “description” element, zero or more “name” elements and an optional “gml:id” attribute.
- The “\_Geometry” element is abstract.

#### D.2.2.3 Geometry primitive

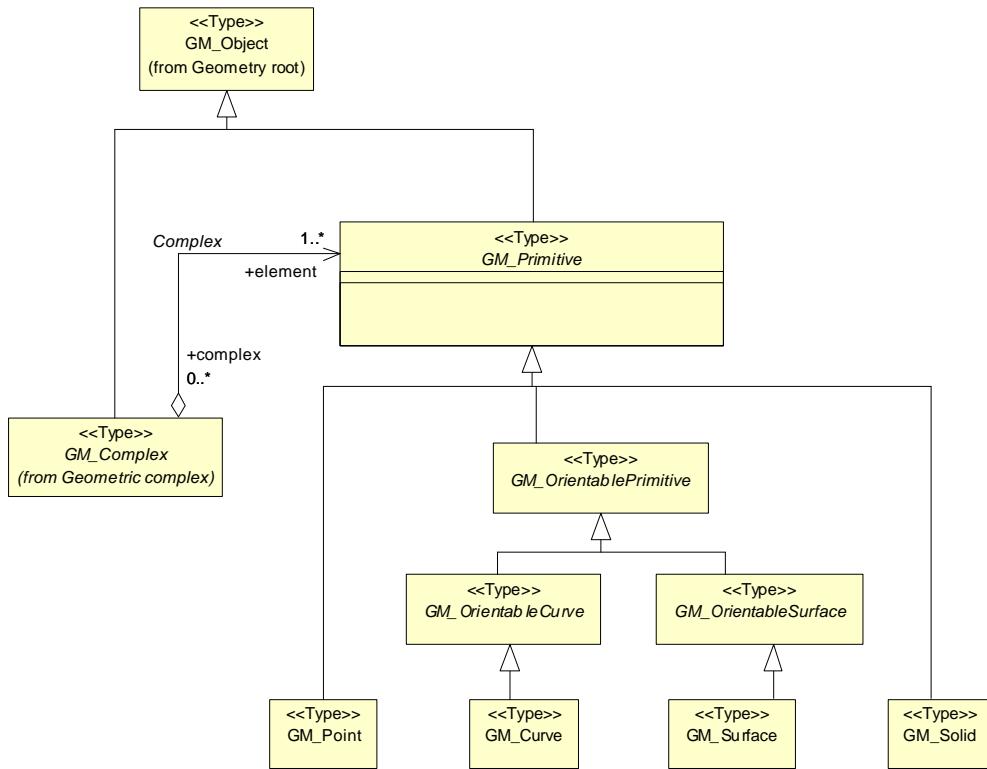
The following UML class diagrams illustrate the profile of the “Geometry root” package (compare with figures 7 to 13 of ISO 19107).

**Figure 15**

The boundary classes from ISO 19107 are not represented explicitly in GML. In ISO 19107 the boundary types are usually the return value of an operation “boundary()”. As the boundary of all surface (patches) or solids needs to be represented in GML explicitly as properties, the exterior and interior properties have been defined in GML directly as properties of the surface (patch) or solid.

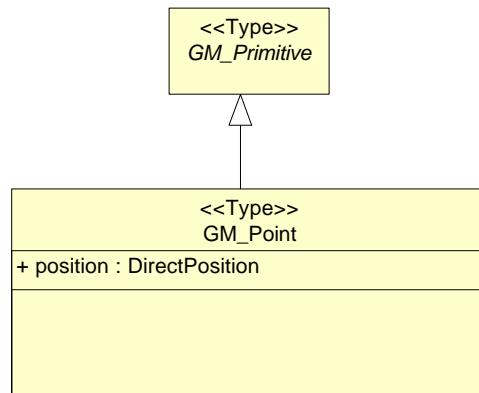
“GM\_Ring” is represented by the “Ring” object element. While a “Ring” is not substitutable for a “CompositeCurve” in GML it is structurally identical to a composite curve.

“GM\_Shell” is also not explicitly represented in GML by its own object element, but a “\_Surface” object element (which can be composite surface) is used instead.



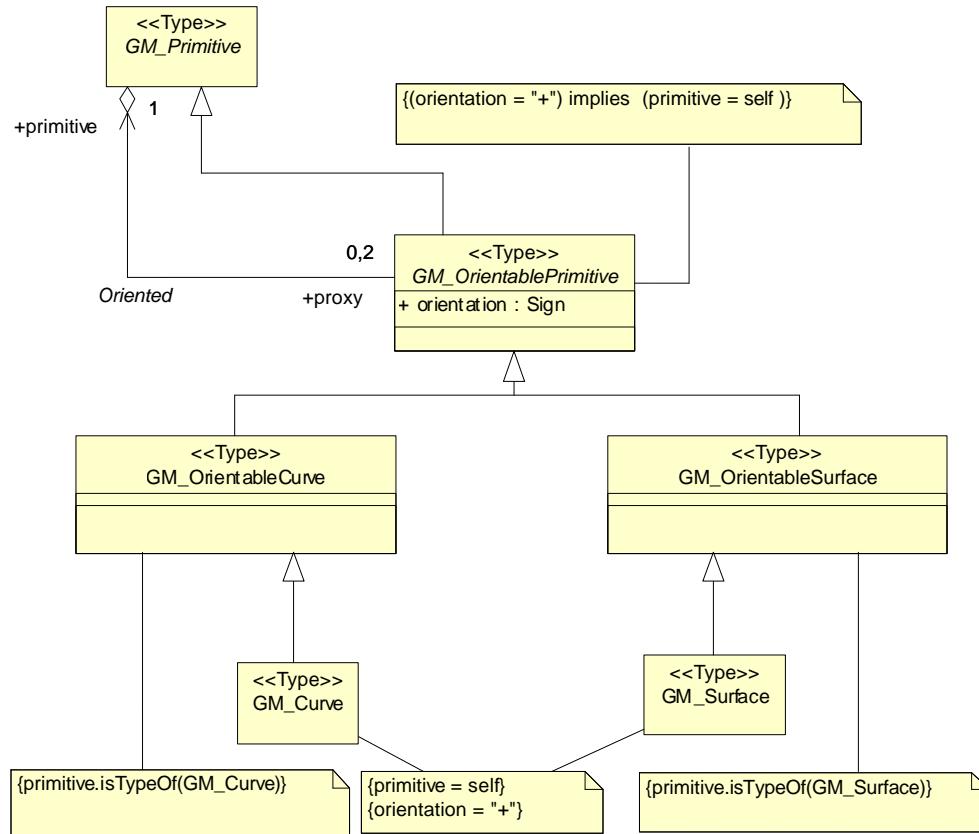
**Figure 16**

“GM\_Primitive” is represented by the “\_GeometricPrimitive” object element (both are abstract). The “complex” role is not navigable in GML.



**Figure 17**

“GM\_Point” is represented by a “Point” object element in GML. The “position” attribute can alternatively be represented by a “pos” property (the type of the value is “DirectPosition”) or a “coordinates” property (the type of the value is “Coordinates”, see below).

**Figure 18**

“GM\_Curve” is represented by the “Curve” object element in GML. The orientation is not an explicit property of a “Curve” and is implicitly fixed to “+”.

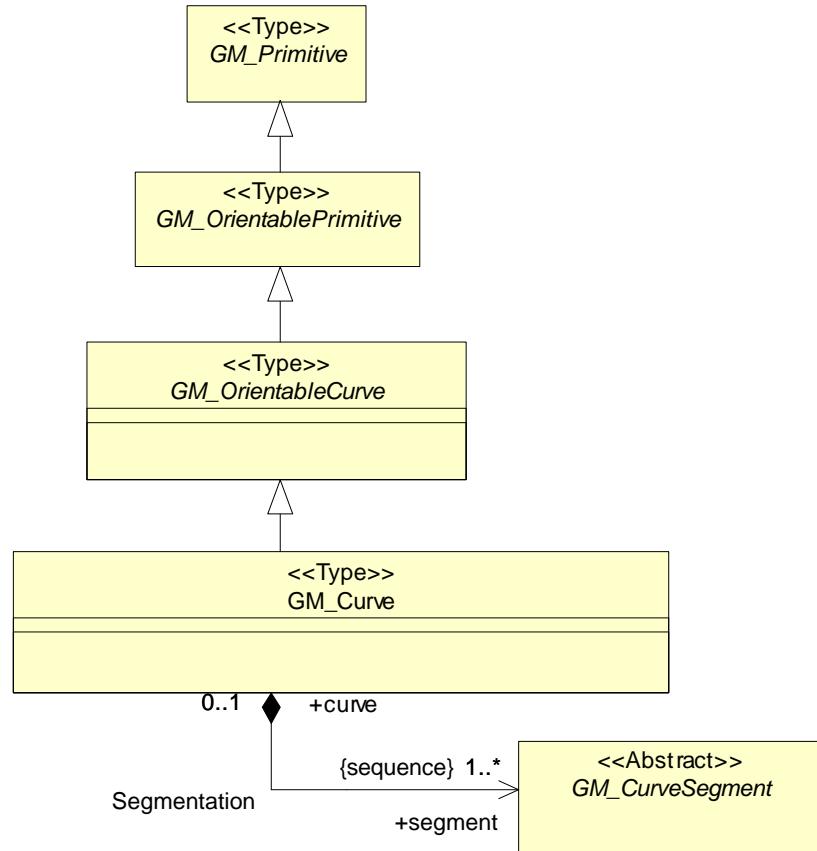
“GM\_OrientableCurve” is represented by the “OrientableCurve” object element in GML. The “primitive” role is represented by the “baseCurve” property.

“GM\_Surface” is represented by the “Surface” object element in GML. The orientation is not an explicit property of a “Surface” and is implicitly fixed to “+”.

“GM\_Orientable Surface” is represented by the “OrientableSurface” object element in GML. The “primitive” role is represented by the “baseSurface” property.

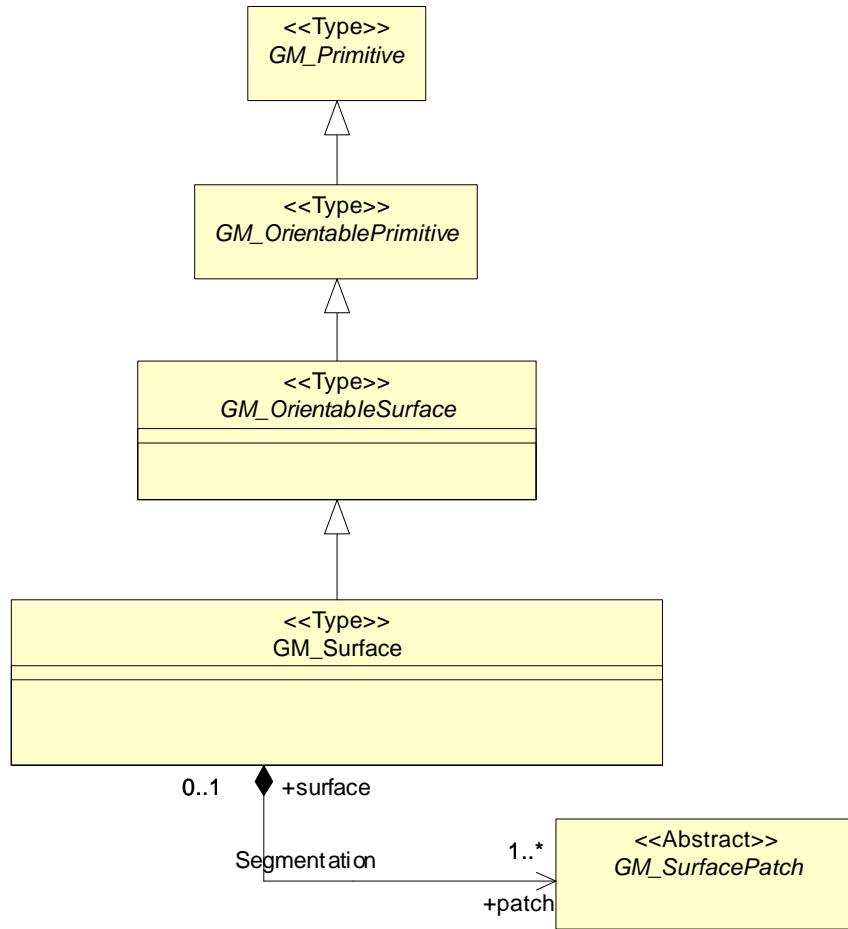
To enable that “CompositeCurve” may be used in GML where in general a geometric primitive is expected, an abstract (and propertyless) object element “\_Curve” has been introduced and can be substituted by either “Curve”, “OrientableCurve” or “CompositeCurve”. The same mechanism is used with surfaces and solids. As a result, the “GM\_OrientablePrimitive” class is not mapped to GML explicitly, however as this type is not instantiable, this does not impose any restrictions.

**NOTE** This mapping is a consequence of the fact that the spatial schema uses multiple inheritance to express that a composite geometry, which by definition is a complex geometry, can also be viewed as a geometric primitive. Since XML Schema is not capable of multiple inheritance (or more precisely: derivation from multiple types), the abstract object elements “\_Curve”, “\_Surface” and “\_Solid” have been introduced in GML to allow that both “true” geometric primitives (e.g. “Curve”) and composite geometries (e.g. “CompositeCurve”) can be in a common substitution group, although both are structurally different.

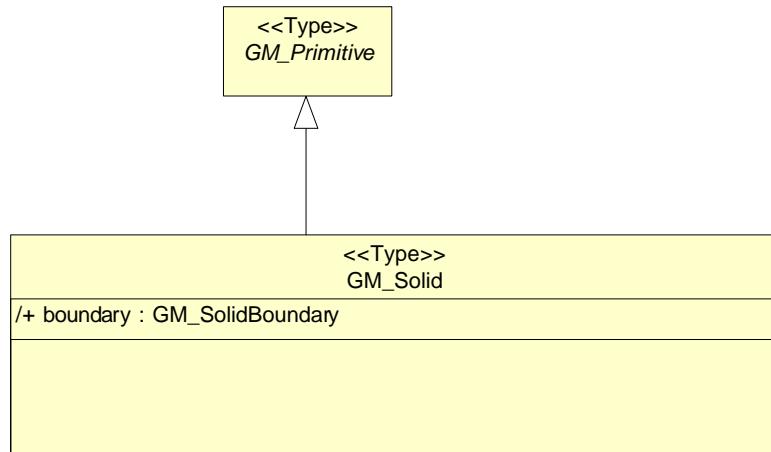


**Figure 19**

As discussed above, “GM\_Curve” is represented by the “Curve” object element in GML. The “segment” role is represented as an array property “segments” in GML.

**Figure 20**

As discussed above, “GM\_Surface” is represented by a “Surface” object element in GML. The “patch” role is represented as an array property “patches” in GML.

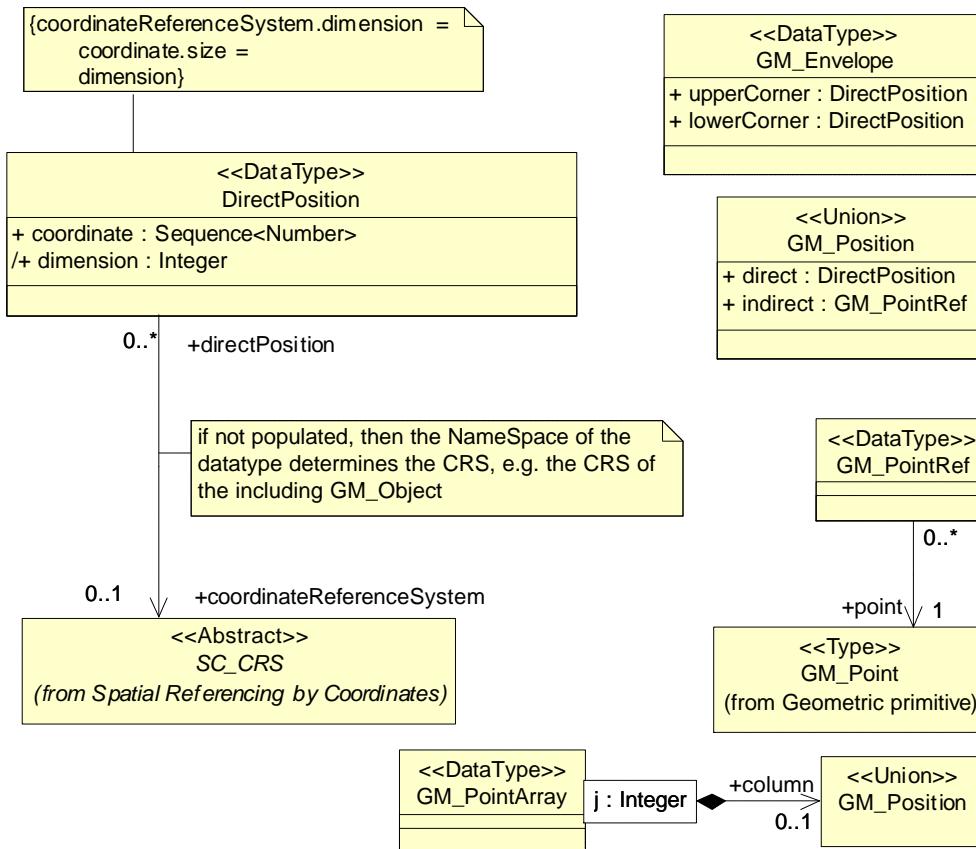


**Figure 21**

“GM\_Solid” is represented by a “Solid” object element in GML. The boundary of a Solid is directly expressed by “exterior” and “interior” properties of the solid as discussed above.

#### D.2.2.4 Coordinate Geometry

The following UML class diagrams illustrate the profile of the Coordinate Geometry package (compare with figures 14 to 21 of ISO 19107).



**Figure 22**

“DirectPosition” is in GML represented as a type with simple content where the “coordinate” attribute is mapped to a list of doubles. The “coordinateReferenceSystem” role is represented by a “srsName” attribute property and “dimension” is represented by an optional attribute property of the same name (type is positiveInteger).

“GM\_Position” is mapped to a choice between a “pos” property (which is of type “DirectPosition”) and a “pointProperty” property (which is “Point”-valued). A “GM\_PointArray” is represented as a similar choice element, but with appropriate settings for minimum and maximum occurrences.

A single “GM\_Position” or a “GM\_PointArray” can alternatively be represented by a “coordinates” property (the type of the value is “Coordinates” which is a type with simple content that represents a list of coordinates encoded as a string).

“GM\_Envelope” is represented as the “Envelope” object element in GML. The two attributes “upperCorner” and “lowerCorner” are mapped to properties of the same name.

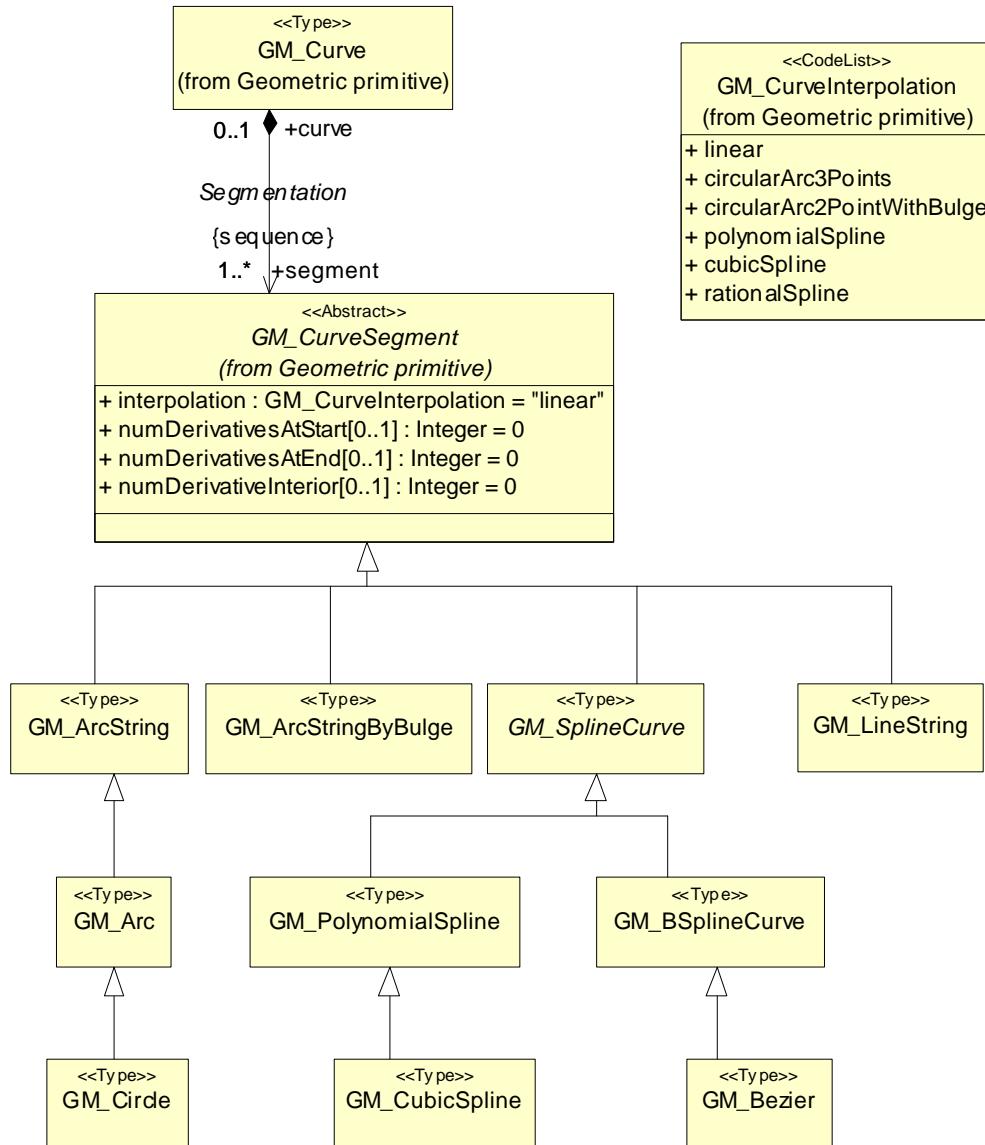


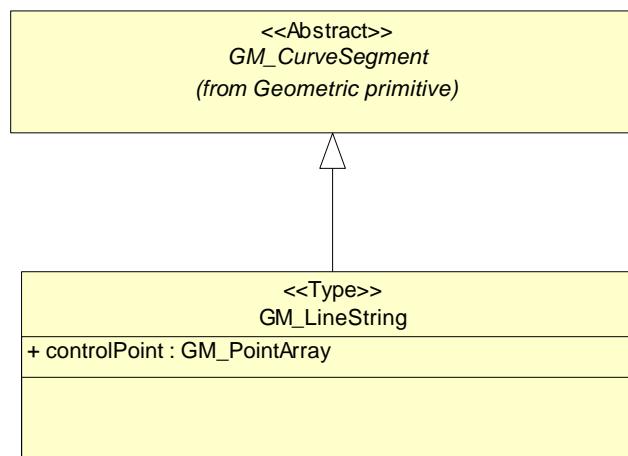
Figure 23

“GM\_CurveSegment” is represented in GML by the “\_CurveSegment” object element (both are abstract). The three “numDerivatives...” attributes are mapped to properties with the same definition. The “interpolation” attribute is not defined in “\_CurveSegment”, but is defined (and set with appropriate initial values) in the instantiable subtypes.

GML currently supports a subset of all defined curve segments of ISO 19107.

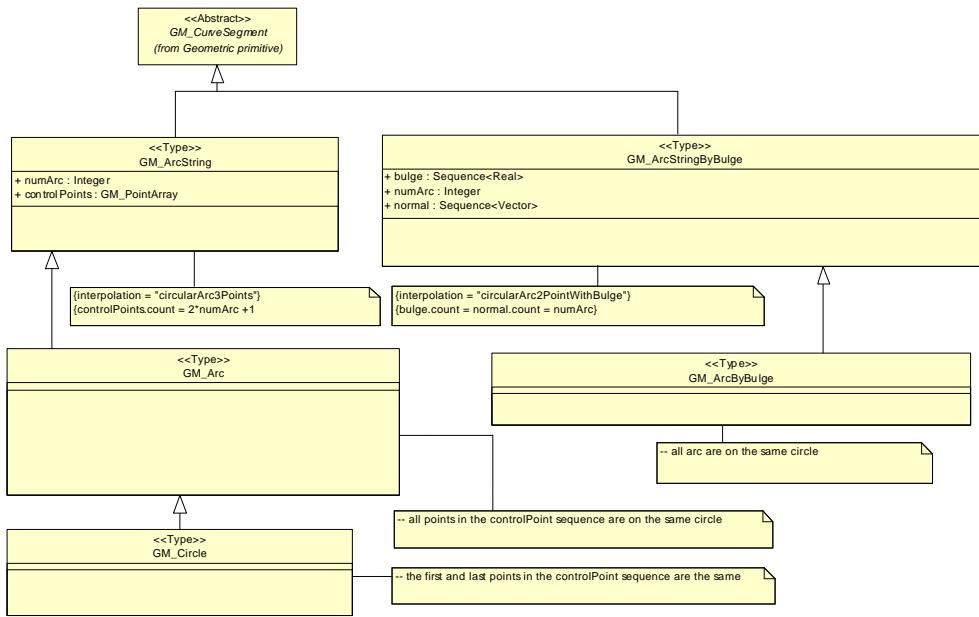
Most subtypes of “GM\_CurveSegment” carry a “controlPoint” attribute that is represented in GML by the choice element as described above (see discussion of the representation of a GM\_PointArray).

The code list “GM\_CurveInterpolation” has been mapped to GML as if it would be an enumeration, i.e. no additional values are allowed beside the predefined values in the GML Schema.

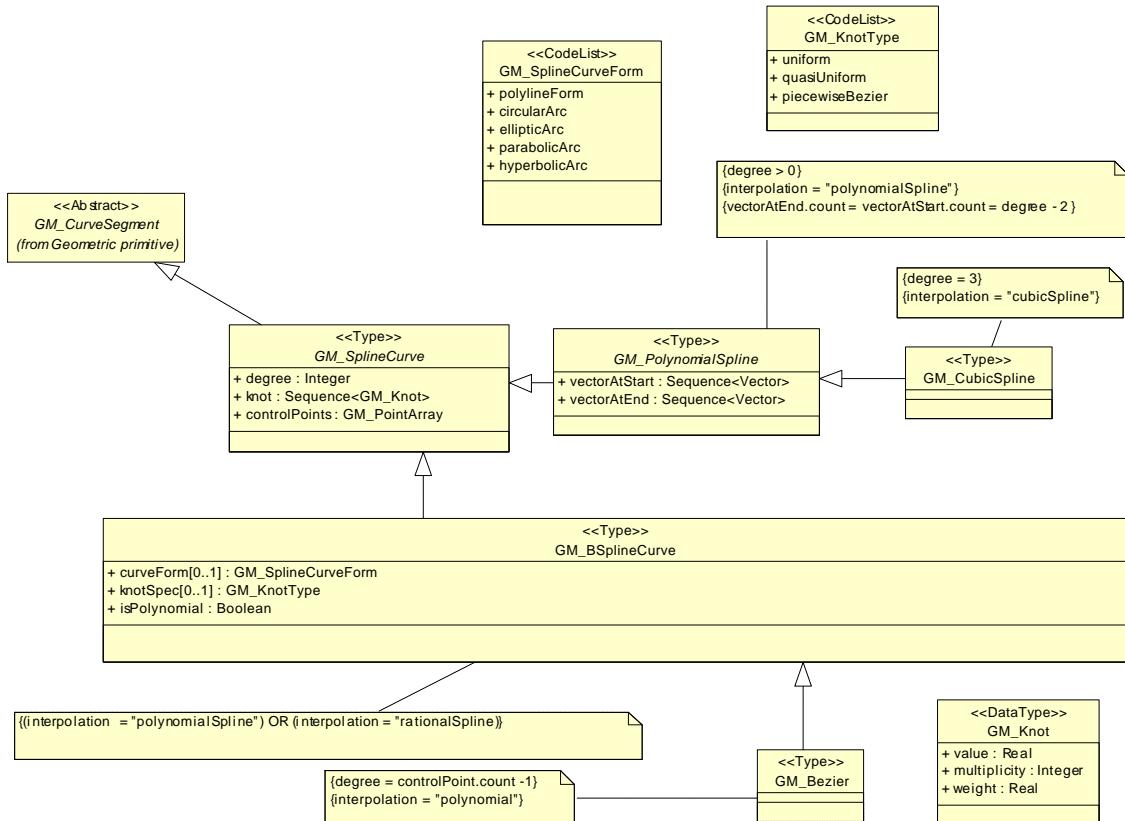


**Figure 24**

“GM\_LineString” is represented by the “LineStringSegment” object element. The “Segment” suffix is appended to the name in GML, because the name “LineString” is already reserved for another object element in GML (see below).

**Figure 25**

The curve segment types are mapped to object elements in GML with the same name (but without the “GM\_” prefix) and the same set of properties.



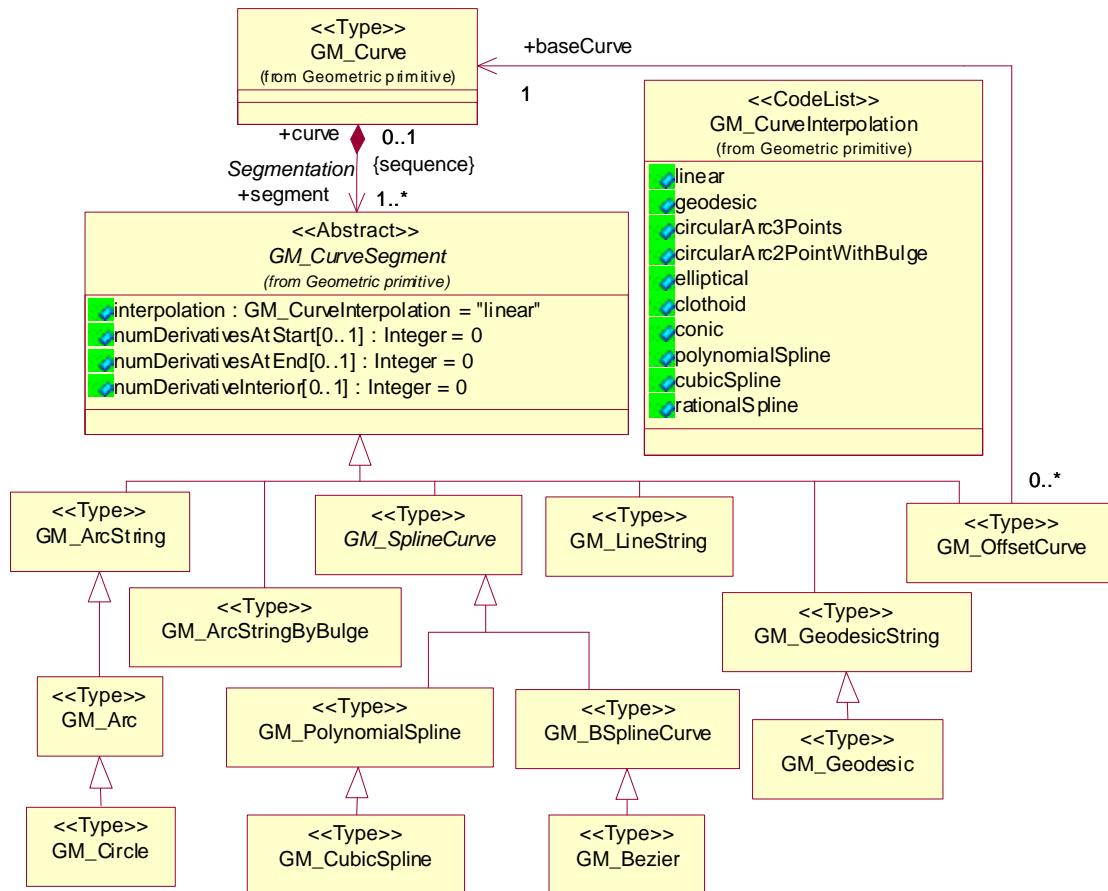
**Figure 26**

Again, these curve segment types are mapped to object elements in GML with the same name (but without the “GM\_” prefix<sup>3</sup>) and the same properties<sup>4</sup>. The properties of the curve segment objects in GML have been specified taking the OCL constraints into account.

The code list “GM\_KnotType” has been mapped to GML as if it would be an enumeration, i.e. no additional values are allowed beside the predefined values in the GML Schema.

<sup>3</sup> However, “GM\_BsplineCurve” is represented by “Bspline”, i.e. without the “Curve” suffix.

<sup>4</sup> The “knotSpec” attribute has been renamed to “knotType” in GML.

**Figure 27**

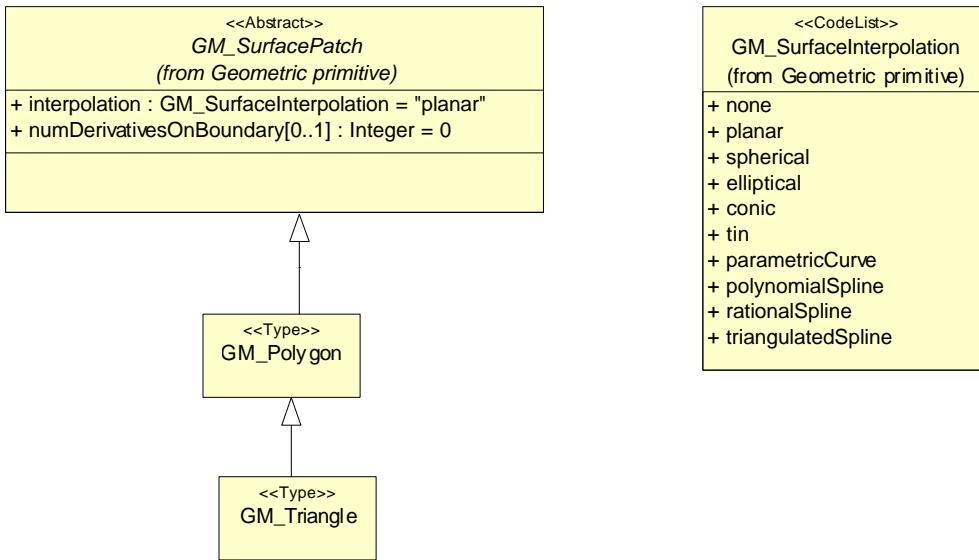
"GM\_OffsetCurve" class is represented in GML by the "OffsetCurve" object element. The object carries the same semantic interpretation as the class. The baseCurve property has been renamed to offsetBase.

"GM\_AffinePlacement" is represented in GML by the "AffinePlacement" object element.

"GM\_GeodesicString" is represented in GML by the "GeodesicString" object element.

"GM\_Geodesic" is represented in GML by the "Geodesic" object element.

"GM\_Clothoid" is represented in GML by the "Clothoid" object element.

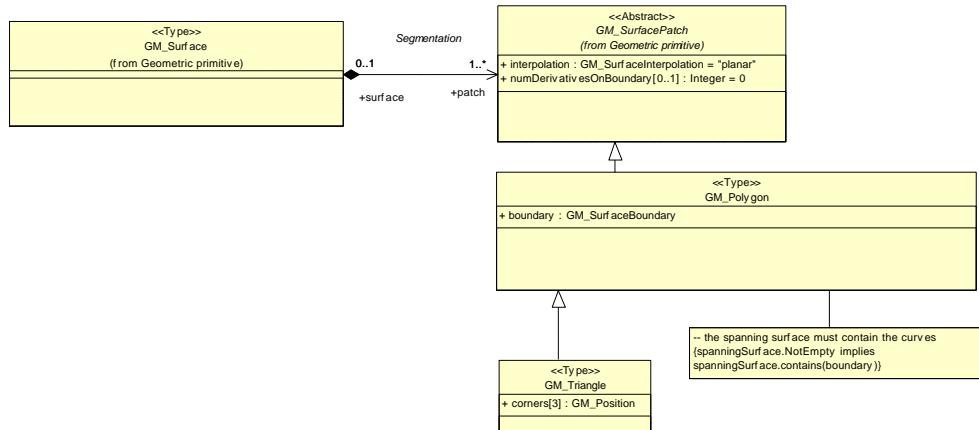


**Figure 28**

“GM\_SurfacePatch” is represented in GML by the “\_SurfacePatch” object element (both are abstract). The “numDerivativesOnBoundary” attribute is currently not explicitly mapped in GML as only planar interpolation is currently supported in GML. The “interpolation” attribute is not defined in “\_SurfacePatch”, but it is defined (and set with appropriate initial values) in the instantiable subtypes.

GML currently supports a subset of all defined surface types and surface patch types of ISO 19107.

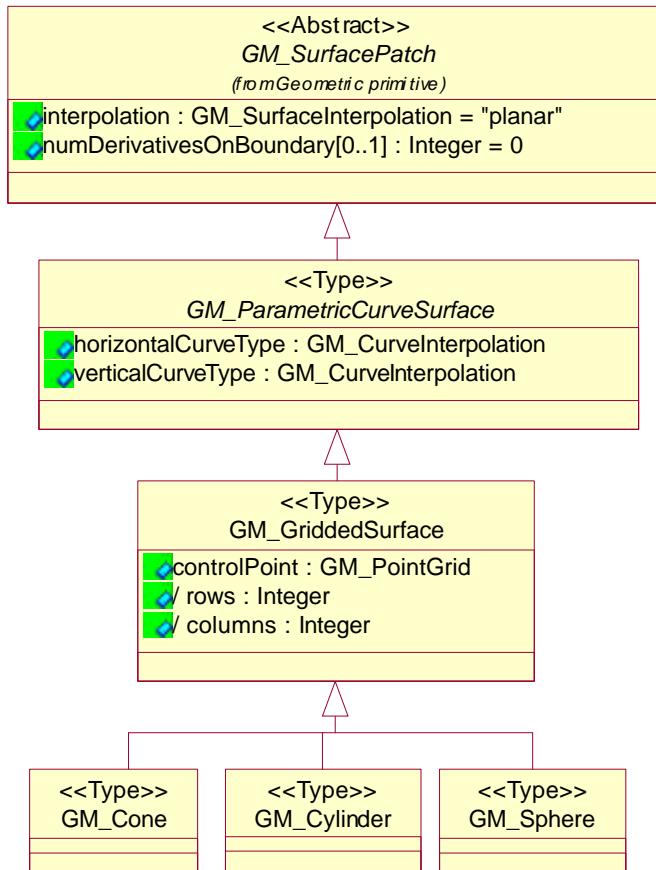
The code list “GM\_SurfaceInterpolation” has been mapped to GML as if it would be an enumeration, i.e. no additional values are allowed beside the predefined values in the GML Schema.



**Figure 29**

“GM\_Polygon” is represented by the “PolygonPatch” object element. The “boundary” attribute is directly expressed by “exterior” and “interior” properties of the “PolygonPatch”.

The “Patch” suffix has been appended to the name in GML, because the name “Polygon” is already reserved for another object element in GML (see below).



**Figure 30**

“GM\_PointGrid” is represented in GML by the “PointGrid” group.

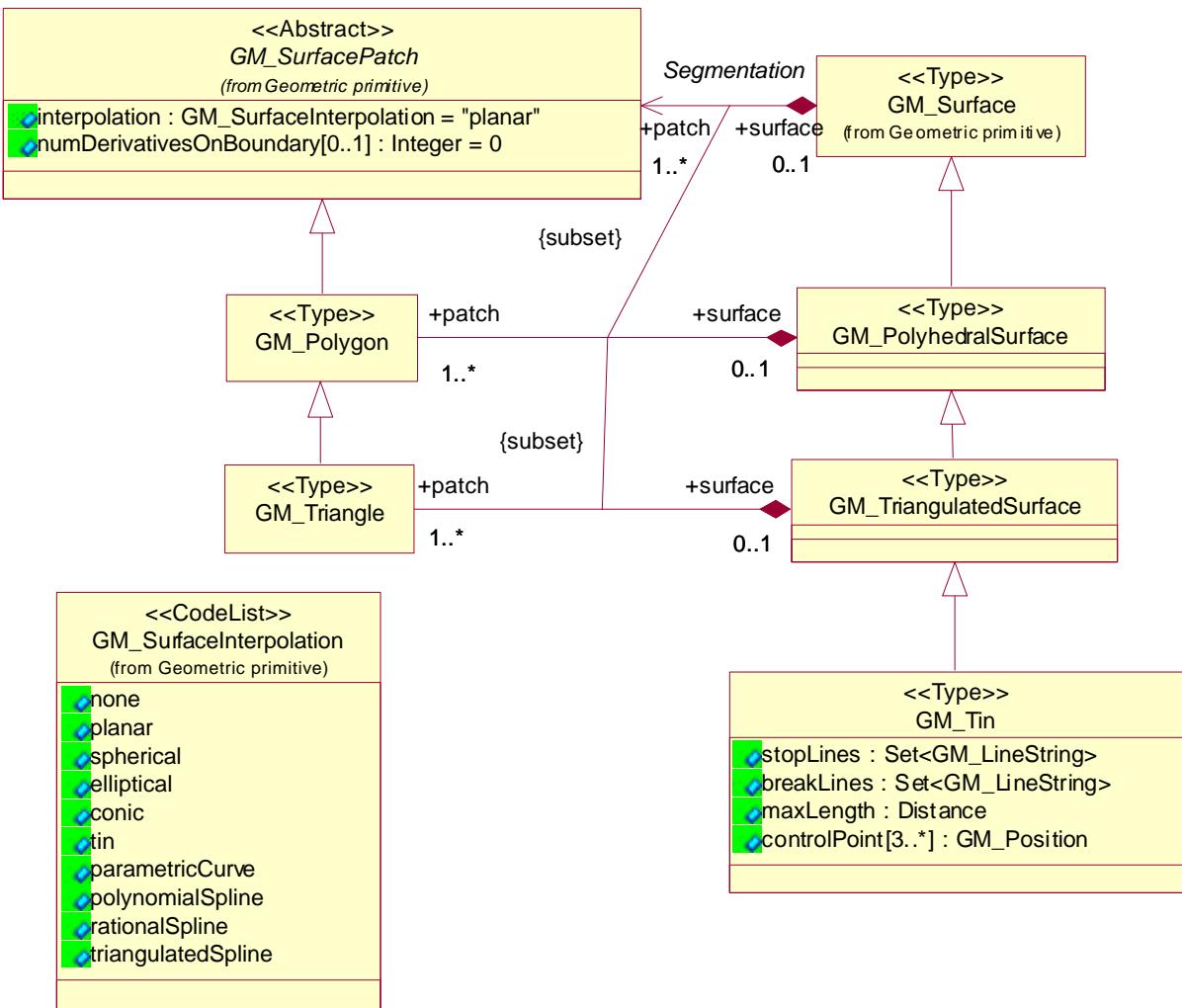
“GM\_ParametricCurveSurface” is represented in GML by the “\_ParametricCurveSurface” object element (both are abstract).

“GM\_GridedSurface” is represented in GML by the “\_GriddedSurface” object element (both are abstract).

“GM\_Cone” is represented in GML by the “Cone” object element.

“GM\_Cylinder” is represented in GML by the “Cylinder” object element.

“GM\_Sphere” is represented in GML by the “Sphere” object element.



**Figure 31**

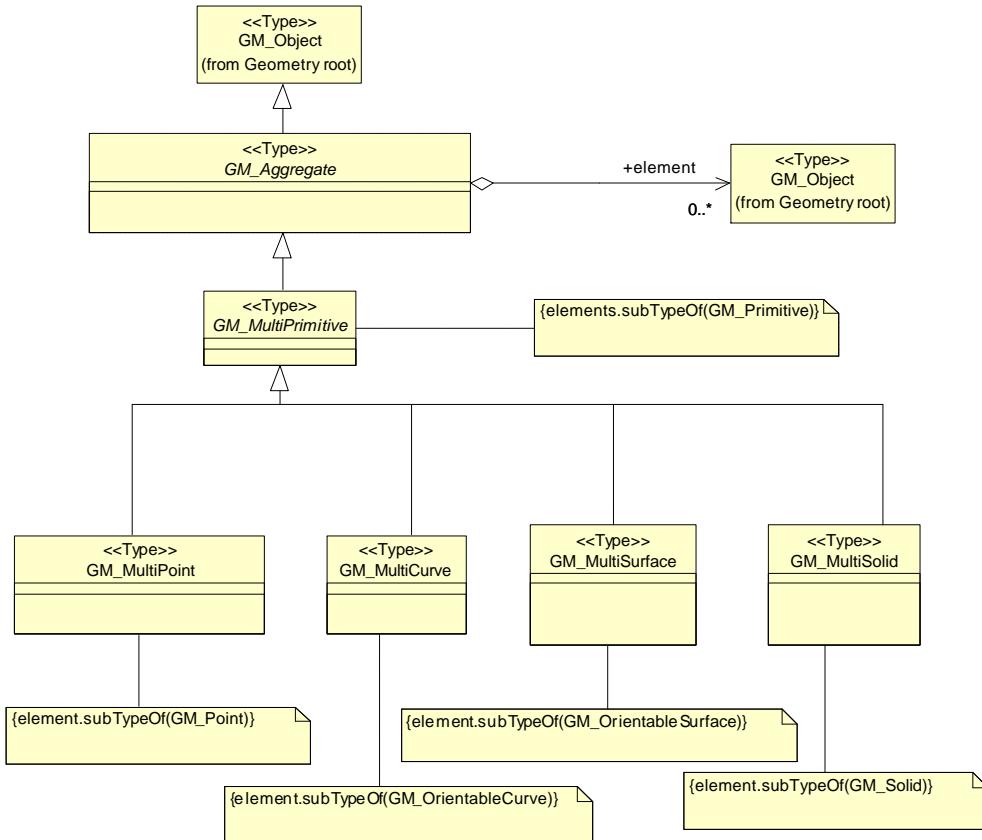
“GM\_PolyhedralSurface” is represented in GML by the “PolyhedralSurface” object element.

“GM\_TriangulatedSurface” is represented in GML by the “TriangulatedSurface” object element.

“GM\_Tin” is represented in GML by the “Tin” object element.

#### D.2.2.5 Geometry aggregates

The following UML class diagrams illustrate the profile of the “Geometry root” package (compare with figure 24 of ISO 19107).

**Figure 32**

“GM\_Aggregate” is represented by the “\_GeometricAggregate” object element (both are abstract). The “element” role is in GML instantiated in the instantiable subtypes. The general pattern is that two properties are defined, one is a regular association property and the other an array association property. The property names are “xMember” and “xMembers” respectively where the “x” is replaced by “point”, “curve”, “surface” or “solid” depending on the elements of the collection. This represents the OCL-constraints for type safetyness.

“GM\_MultiPoint” is represented by the “MultiPoint” object element in GML.

“GM\_MultiCurve” is represented by the “MultiCurve” object element in GML.

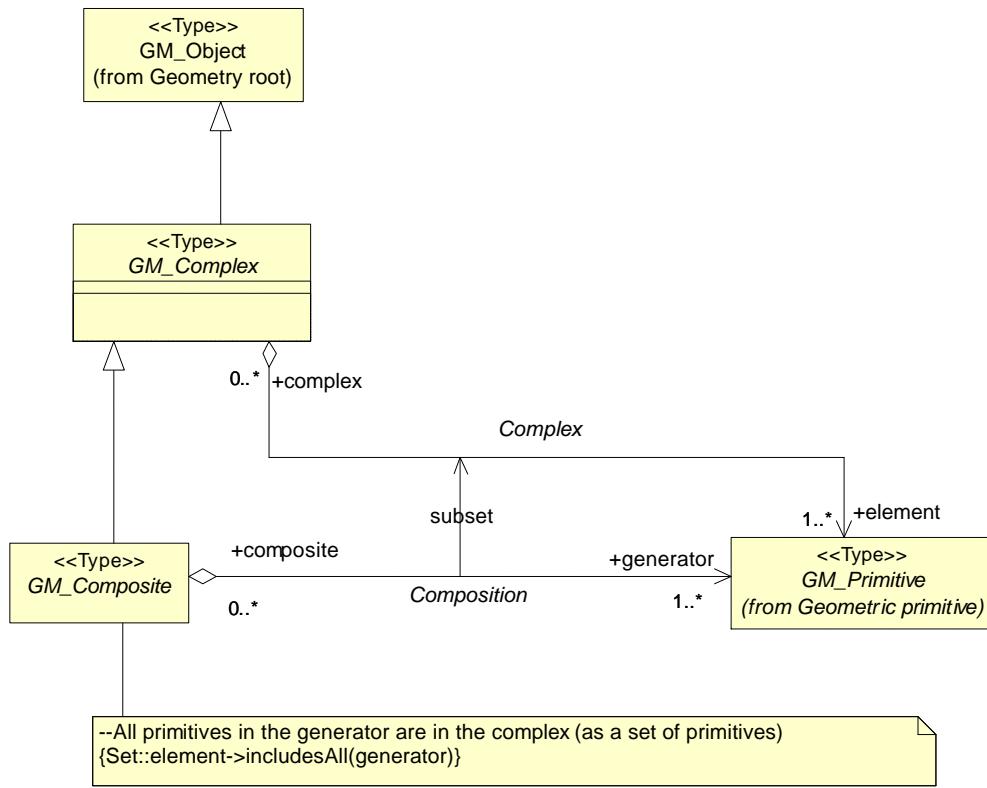
“GM\_MultiSurface” is represented by the “MultiSurface” object element in GML.

“GM\_MultiSolid” is represented by the “MultiSolid” object element in GML.

“GM\_MultiPrimitive” is not explicitly represented in GML.

#### D.2.2.6 Geometry complex

The following UML class diagrams illustrate the profile of the “Geometry root” package (compare with figures 25 to 30 of ISO 19107).

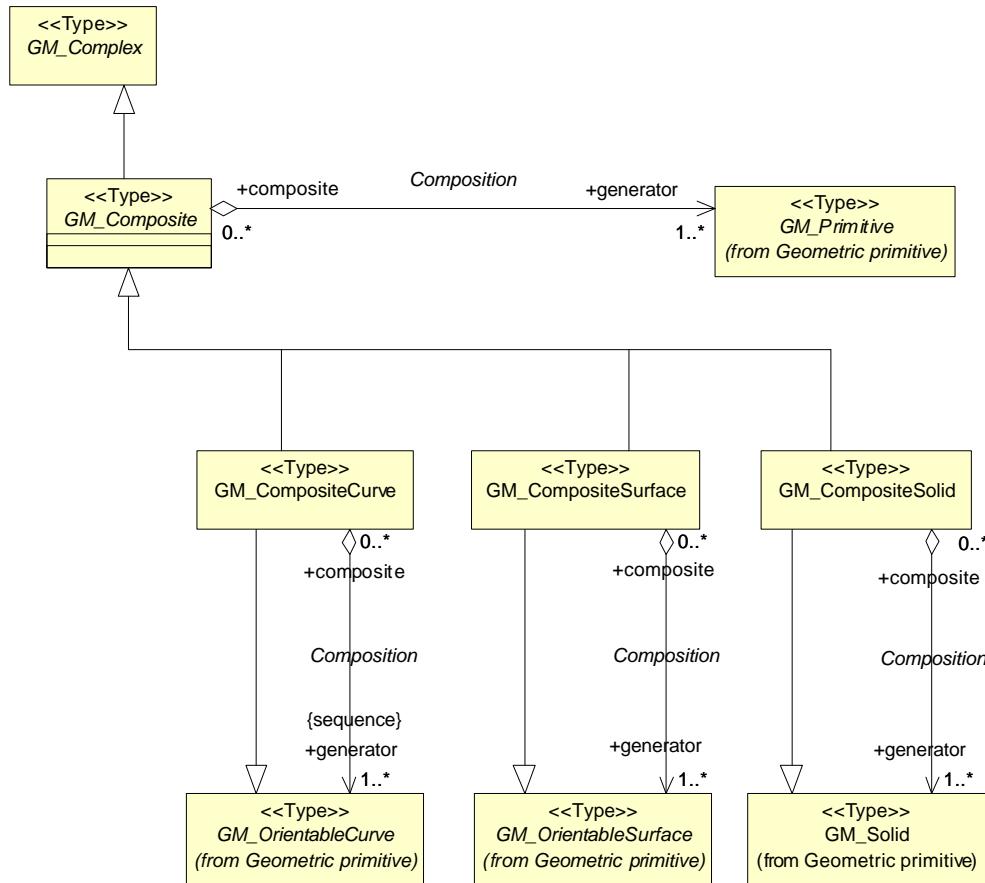


**Figure 33**

“GM\_Complex” is represented by the “GeometricComplex” object element. The “element” role is mapped to a property of the same name in GML.

“GeometricComplex” can be instantiated (unlike “GM\_Complex” which is abstract).

The fact that the composite geometry types are subtypes of “GM\_Complex” is represented in GML by the fact that every association property which takes a “GeometricComplex” accepts also one of the composites (due to a choice element in “GeometricComplexPropertyType”). This special mapping to XML Schema was necessary, because multiple inheritance, used in ISO 19107 to express the “dualism” of the composite geometries, is not supported by the derivation mechanism of XML Schema.

**Figure 34**

"GM\_Composite" is not explicitly represented by an object element in GML. However, the subtypes "GM\_CompositeCurve", "GM\_CompositeSurface" and "GM\_CompositeSolid" are represented in GML by object elements of the same name (without the "GM\_" prefix). The "generator" role is in GML instantiated in these subtypes by an association property with the name "xMember" where the "x" is replaced by "curve", "surface" or "solid" depending on the elements of the collection.

#### D.2.2.7 Conformance

The rules governing conformance of a profile of ISO 19107 are described in chapter 2 and annex A of ISO DIS 19107. Concerning the three criteria defined in chapter 2, GML geometry covers the following levels:

Data Complexity:

- Geometric primitives
- Geometric complexes

Dimensionality:

- 0-, 1-, 2- and 3-dimensional objects

Functional Complexity:

- Data types only

Thus, the relevant conformance clauses of ISO 19107 are:

- A.1.1.1 - A.1.1.4
- A.2.1.1 - A.2.1.3

The conditions of these conformance clauses are met.

Note that derived attributes are treated as operations and it is assumed that the derived attributes in the aggregate geometries will be derived from the data by the application handling the GML instances.

A GM\_CompositePoint is represented by a “Point” object element in GML. The value of the “generator” association role is the same object, i.e. the “Point” object itself.

### D.2.3 ISO 19107 Spatial Schema (Topology)

#### D.2.3.1 Overview

The following additional changes have been applied to the topology package of ISO 19107.

**Table 11**

Change	Explanation
TP_Complex: association „isMaximal()“ added as a derived attribute	The information was otherwise not accessible by means of predefined data structures of TP_Complex. The attribute shall be defined as a derived attribute representing the result of the “isMaximal()” operation as defined in ISO 19107.
TP_Object has been changed from an interface class to type class (however without any properties) and the Realization relationships from TP_Primitive and TP_Complex to TP_Complex has been changed to Specialization relationships.	Maintaining TP_Object as a root for the different topological subtypes makes its mapping to GML clearer.
The optional association “Realization” between TP_Complex and GM_Complex has been deleted.	The realization can be derived from the realization of the primitives contained in the topological complex.
The “maximalComplex” role has been deleted from TP_Primitive.	Currently not supported in GML
TP_Boundary and subtypes as well as TP_Ring and TP_Shell have been deleted.	Only used in operations

#### D.2.3.2 Topology root

The following UML class diagrams illustrate the profile of the “Topology root” package (compare with figures 32 to 33 of ISO 19107).

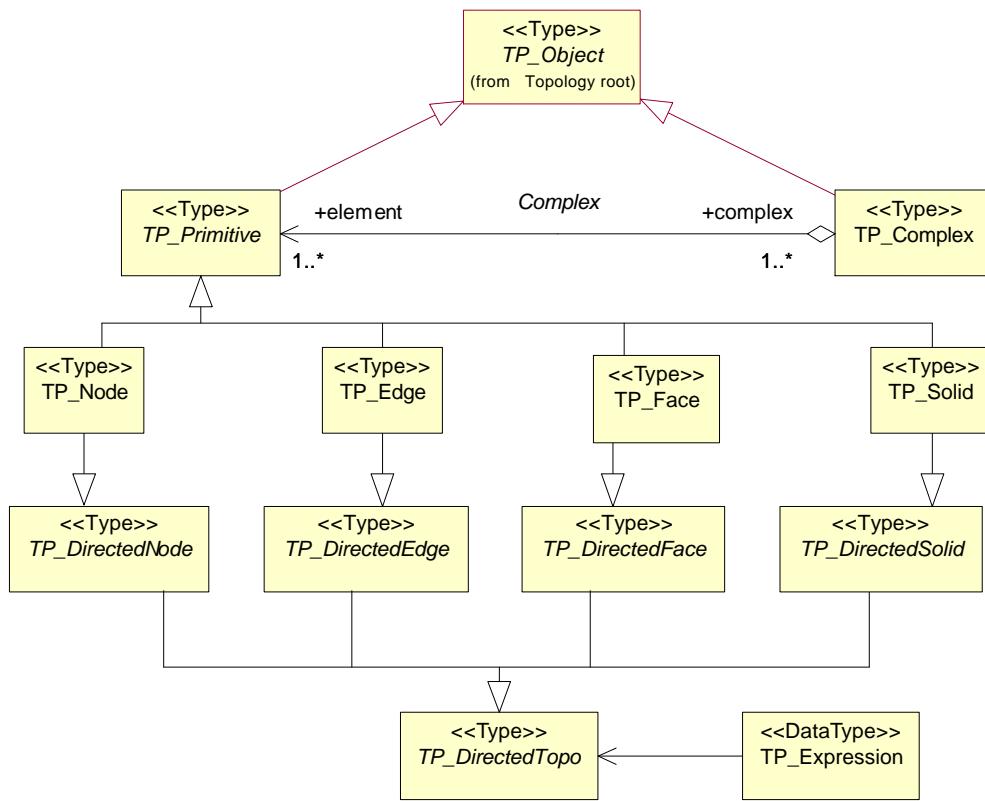
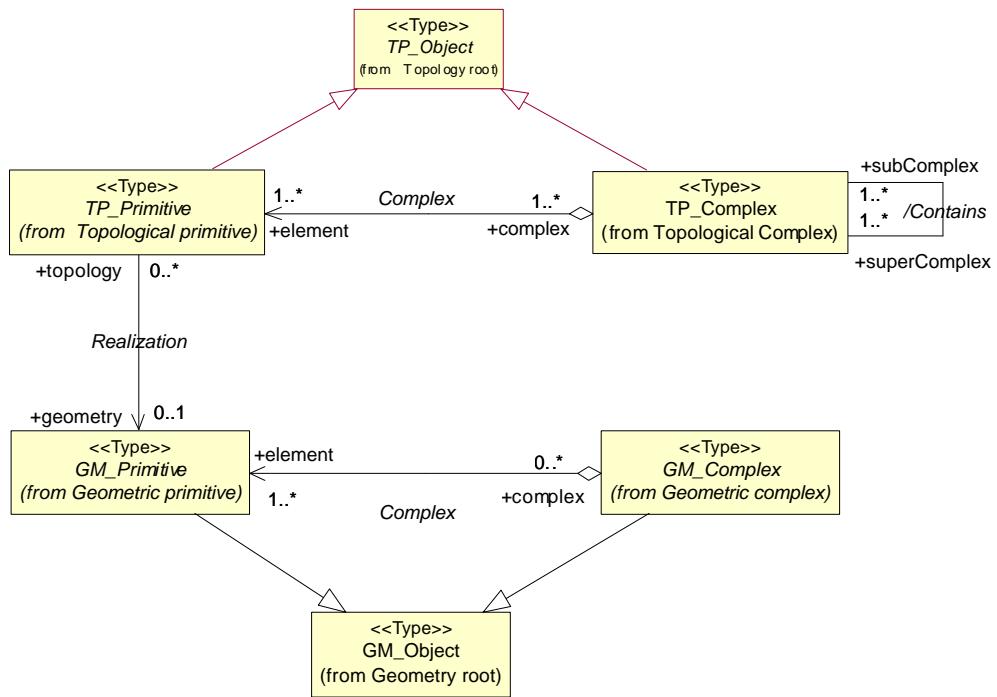


Figure 35



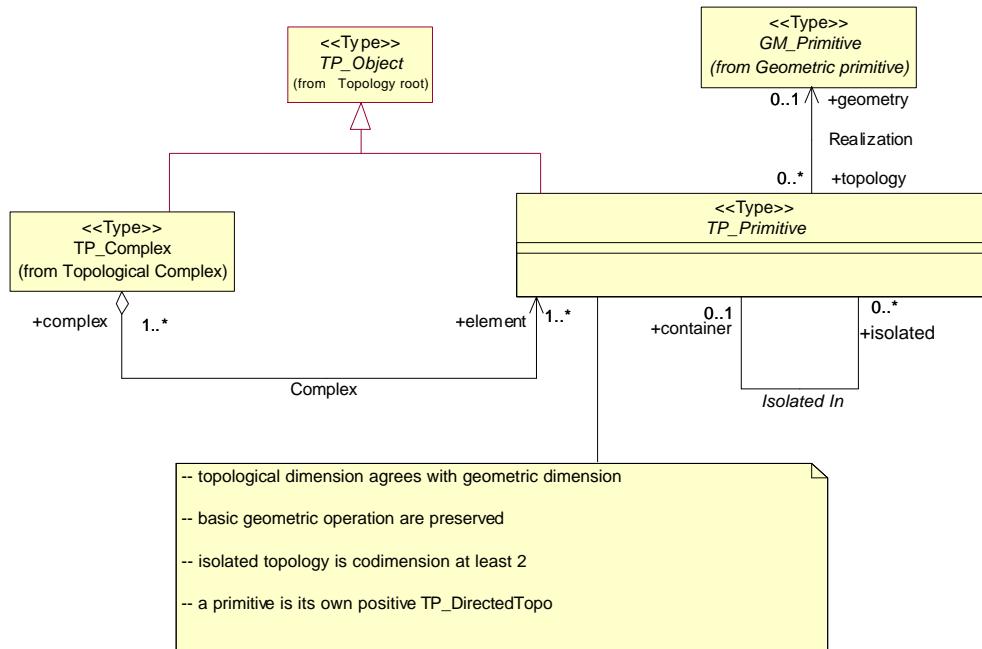
**Figure 36**

The mapping of the different classes to the GML Schemas is explained in the subsequent sections showing details of the class hierarchy.

"TP\_Object" is represented by the "\_Topology" object element. The "\_Topology" element may carry additional properties: an optional "description" element, zero or more "name" elements and an optional "gml:id" attribute.

#### D.2.3.3 Topology primitive

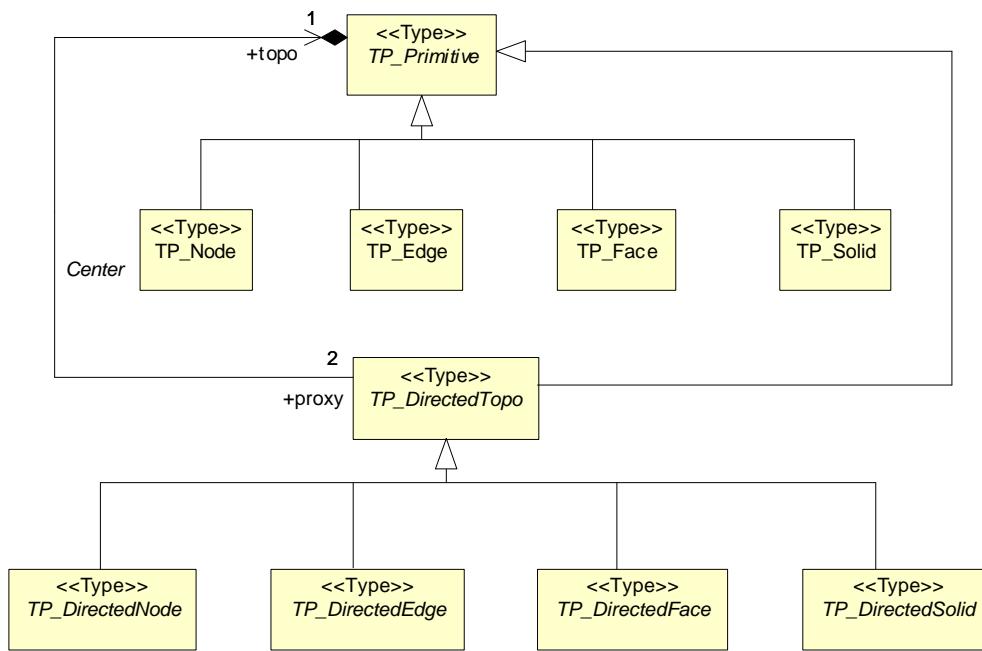
The following UML class diagrams illustrate the profile of the "Topology root" package (compare with figures 35 to 45 of ISO 19107).

**Figure 37**

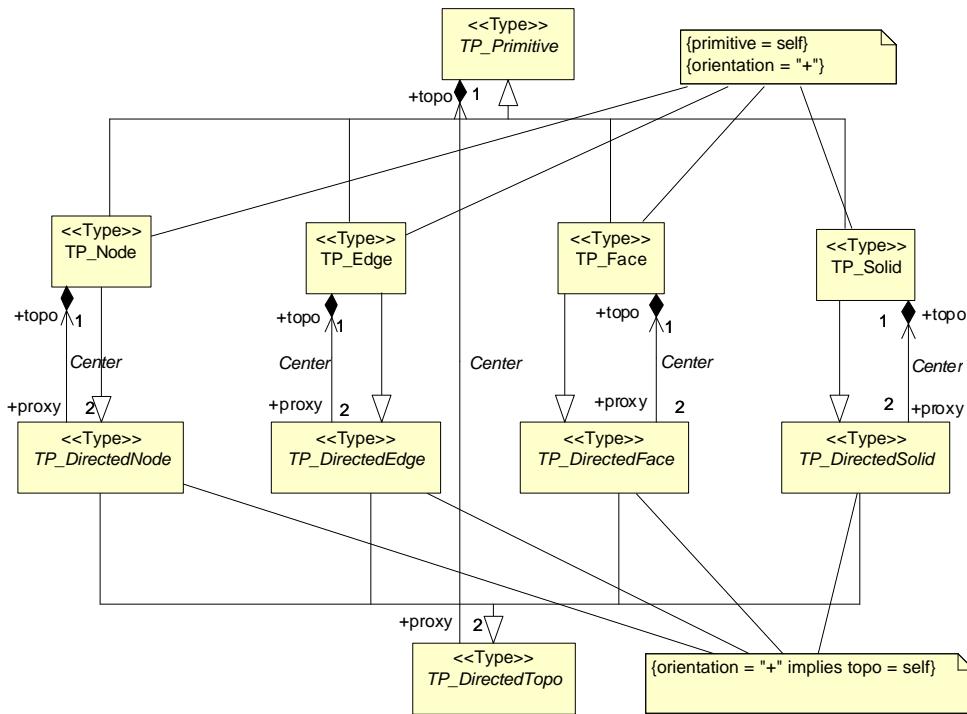
"TP\_Primitive" is represented by the "\_TopoPrimitive" object element.

The "geometry" role is instantiated in the instantiable subtypes. This allows to control the geometry types at the other association end (dimensionality constraint): "pointProperty", "curveProperty", "surfaceProperty" and "solidProperty" respectively.

The "isolated" and "container" roles are represented as properties in "\_TopoPrimitive".



**Figure 38**



**Figure 39**

"TP\_Node" is represented by the "Node" object element in GML.

"TP\_Edge" is represented by the "Edge" object element in GML.

"TP\_Face" is represented by the "Face" object element in GML.

"TP\_Solid" is represented by the "TopoSolid" object element in GML (the name "Solid" is already used for the 3-dimensional geometry primitive).

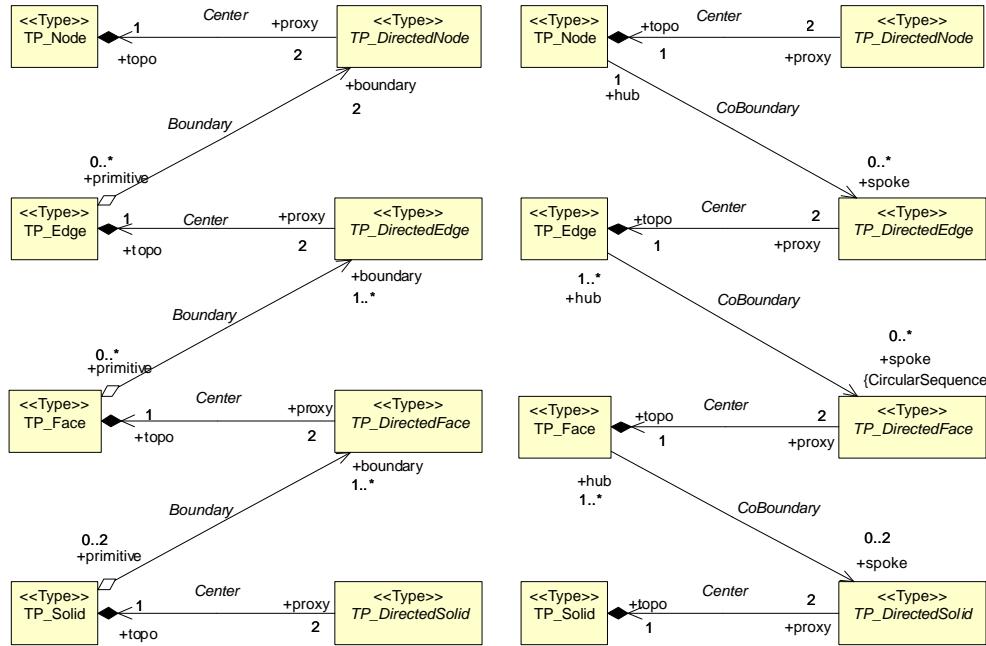
"TP\_DirectedTopo" is not explicitly represented in GML, only its instantiable subtypes. A notable difference is that although the directed topology types are modelled as types they are represented as properties with an "orientation" attribute in GML.

"TP\_DirectedNode" is represented by the "directedNode" property element in GML. The "topo" role is represented directly by the "Node" object element.

"TP\_DirectedEdge" is represented by the "directedEdge" property element in GML. The "topo" role is represented directly by the "Edge" object element.

"TP\_DirectedFace" is represented by the "directedFace" property element in GML. The "topo" role is represented directly by the "Face" object element.

"TP\_DirectedSolid" is represented by the "directedTopoSolid" property element in GML. The "topo" role is represented directly by the "TopoSolid" object element.



**Figure 40**

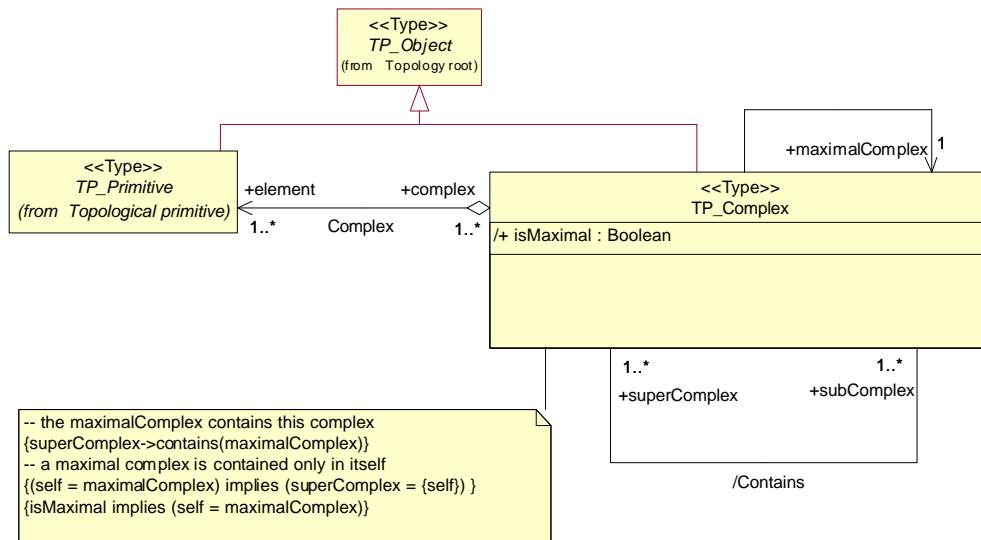
The mapping of the "topo" role is already discussed above.

The "spoke" role is represented in GML by "directed Edge", "directedFace" and "directedTopoSolid" properties respectively.

The “boundary” role is represented in GML by “directed Node”, “directedEdge” and “directedFace” properties respectively.

#### D.2.3.4 Topology complex

The following UML class diagrams illustrate the profile of the “Topology complex” package (compare with figure 46 of ISO 19107).



**Figure 41**

“TP\_Complex” is represented by the “TopoComplex” object element.

The “element” role is mapped to two properties – one regular association property “topoPrimitiveMember” and one array association property “topoPrimitiveMembers”.

The “subComplex” and “superComplex” roles are represented as association properties of the same name. The minimum multiplicity, however, is “0” for both properties in GML instead of “1”. This reflects that it is not required that this property is represented explicitly in a GML instance (note that it is a derived association).

The “maximalComplex” role is represented as an association property of the same name in GML.

#### D.2.3.5 Conformance

The rules governing conformance of a profile with ISO 19107 are described in chapter 2 and annex A of ISO DIS 19107. Concerning the three criteria defined in chapter 2, GML topology covers the following levels:

Data Complexity:

- Topological complexes
- Topological complexes with geometric realizations

Dimensionality:

- 0-, 1-, 2- and 3-dimensional objects

Functional Complexity:

- Data types only

Thus, the relevant conformance clauses of ISO 19107 are:

- A.3.1.1 - A.3.1.3
- A.4.1.1 - A.4.1.3

The conditions of these conformance clauses are met.

Note that the association “Realization” between TP\_Complex and GM\_Complex is not an explicit part of the profile, because the geometrical realization of the topological complex can be derived from the geometrical realization of the topological primitives.

## D.2.4 ISO 19108 Temporal Schema

### D.2.4.1 Overview

The GML temporal schemas provides an implementation of clauses 5.2 - 5.4 of ISO 19108:2002.

The following changes have been applied to the packages of ISO 19108.

**Table 12**

	Change	Explanation
1	The <i>beginning</i> and <i>ending</i> associations are implemented as unidirectional from TM_Period to TM_Instant	Maintaining bi-directional pointers deemed unnecessary in a static data encoding
2	The <i>beginning</i> and <i>ending</i> associations from TM_Period are implemented as a choice of (a UML association with TM_Instant or a UML attribute of type TM_Position)	Allows a more compact encoding of the same information when a TM_Instant object with identity is not required
3	The <i>termination</i> and <i>initiation</i> associations are implemented as unidirectional from TM_Edge to TM_Node	Maintaining bi-directional pointers deemed unnecessary in a static data encoding
4	The <i>realization</i> associations are implemented as unidirectional from TM_Instant and TM_Period to TM_Node and TM_Edge respectively	Maintaining bi-directional pointers deemed unnecessary in a static data encoding
5	The <i>basis</i> association is implemented as unidirectional from TM_Calendar to TM_CalendarEra	Maintaining bi-directional pointers deemed unnecessary in a static data encoding
7	The <i>begin</i> and <i>end</i> attributes of TM_OrdinalEra are replaced by associations with TM_Node, with rolenames <i>start</i> and <i>end</i>	Practice in historical and geological sciences is that the termination points of an Ordinal Era are associated with events whose position may not be known precisely. This matches with the concept of TM_Node whose position is available indirectly.
8	The <i>origin</i> attribute of TM_CoordinateSystem is implemented as choice of (a UML association with TM_Instant or a UML attribute of type TM_Position)	Allows the origin to be specified in terms of an external event.
9	The <i>interval</i> attribute of TM_CoordinateSystem is implemented as TM_Interval Length	Allows the scale to be specified more precisely and flexibly.

NOTE 1 Change 2 takes advantage of the <choice> structure which is provided by the XML Schema implementation language. This supports a more flexible and compact encoding, containing the same information, than would have been gained by mechanical application of the standard encoding rules.

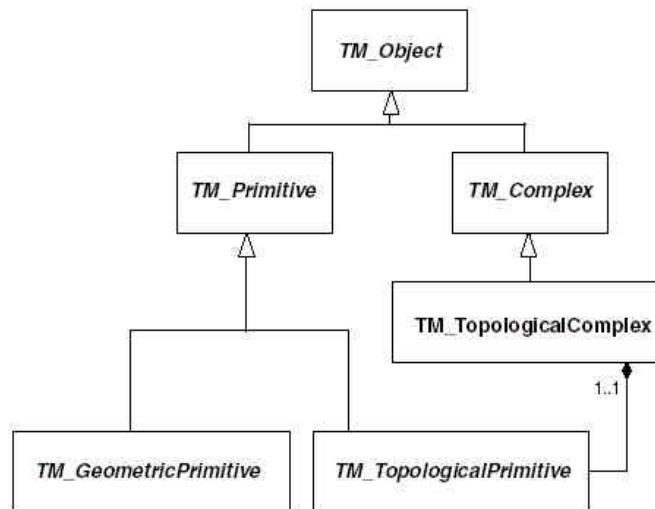
NOTE 2 A change proposal has been submitted regarding ISO 19108 to match the issue identified in change 7.

Of the classes dealing with temporal relationships between features, described in clause 5.5 of ISO 19108:2002, only Feature Succession has been implemented directly. Components corresponding to the other relationships may be defined in GML application schemas, but are not discussed further here.

The mapping of the different classes to the GML Schemas is explained in the subsequent sections showing details of the class hierarchy.

The following UML class diagrams illustrate the profile of the “Temporal Objects” package (compare with figures 2 to 6 and 11 of ISO 19108:2002).

#### D.2.4.2 Temporal Objects



**Figure 42 Main hierarchy of temporal objects from ISO 19108:2002**

“TM\_Object” is represented by the “\_TimeObject” object element. The “\_TimeObject” element can carry additional properties: an optional “description” element, zero or more “name” elements and an optional “gml:id” attribute. These properties are inherited by all the components that are substitutable for \_TimeObject.

“TM\_Primitive” is represented by the “\_TimePrimitive” object element.

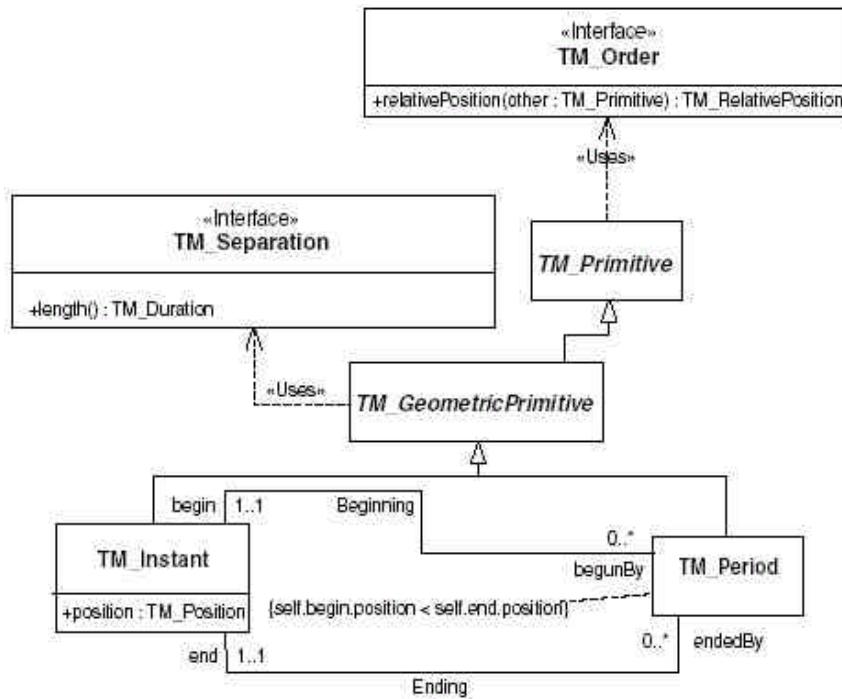
“TM\_GeometricPrimitive” is represented by the “\_TimeGeometricPrimitive” object element.

“TM\_TopologicalPrimitive” is represented by the “\_TimeTopologyPrimitive” object element.

“TM\_Complex” is represented by the “\_TimeComplex” object element.

“TM\_TopoComplex” is represented by the “\_TimeTopologyComplex” object element.

#### D.2.4.3 Concrete Temporal Geometric Primitives



**Figure 43 Profile of temporal geometric objects adapted from ISO 19108:2002**

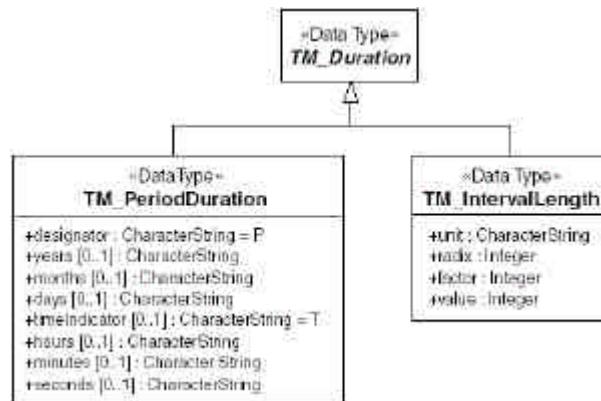
“TM\_Primitive” is represented by the “\_TimePrimitive” object element. Additional properties “relatedTime” representing the result of “relativePosition(other:TM\_Primitive)” operations has been added.

“TM\_GeometricPrimitive” is represented by the “\_TimeGeometricPrimitive” object element. An additional property “\_timeLength” representing the result of the “length()” operation has been added.

“TM\_Instant” is represented by the “TimeInstant” object element. The “position” attribute is represented by the “timePosition” property.

“TM\_Period” is represented by the “TimePeriod” object element. The “begin” and “end” roles are represented by association properties of the same name in GML. These associations have an alternative representation in GML as follows: “end” is in a choice block with “endPosition” and “begin” with “beginPosition”, the latter in each case has simple content as discussed in clause 14.2.3.4.

#### D.2.4.4 Temporal Duration



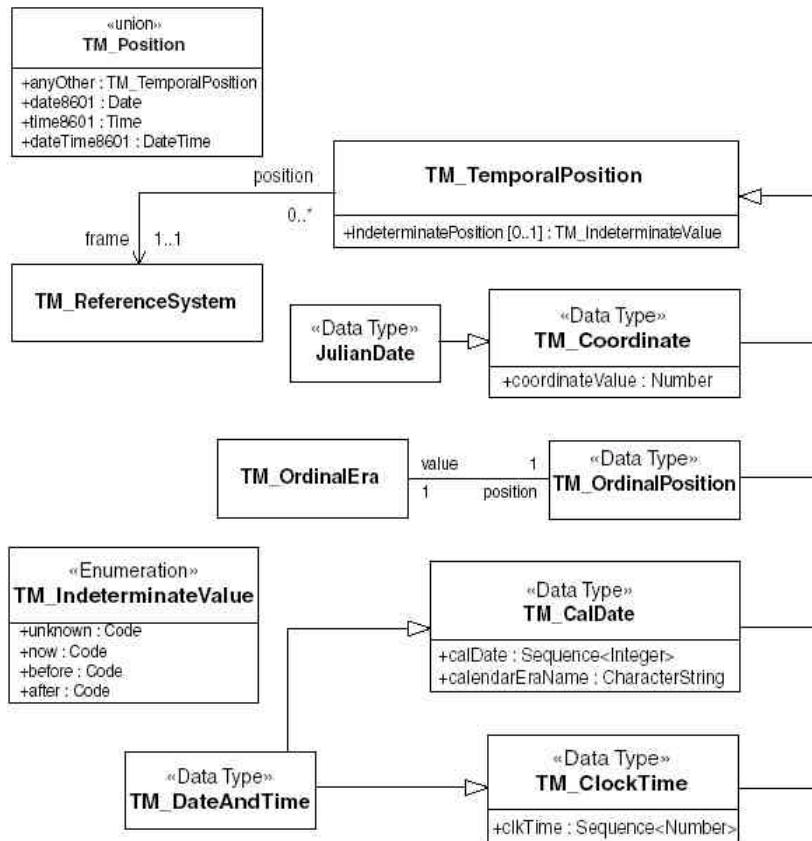
**Figure 44      DataTypes representing temporal duration ISO 19108:2002**

The GML property element “\_timeLength” is abstract, with either an “timeInterval or “duration” element substituting. These have XML Schema types which implement the data types shown in Figure 44, as follows:

“TM\_IntervalLength” is implemented using a simpleContent type constructed by adding the XML attributes “unit”, “radix” and “factor” to the XML Schema built-in type “decimal”;

“TM\_PeriodDuration” is implemented by the XML Schema built-in type “duration” (see discussion in clause 14.2.3.7). The XML Schema type “duration” prescribes a literal value with the lexical form described in ISO 8601, which removes the need to implement the list of attributes of the TM\_PeriodDuration class separately.

#### D.2.4.5 Temporal Position



**Figure 45 Temporal position, from ISO 19108:2002**

Components represented as UML attributes of type **TM\_Position** are represented as GML properties with XML Schema type **gml:TimePositionType**. This is a “simpleContent” type the details of whose derivation are described in clause 14.2.3.4. This represents the requirements shown in Figure 45 as follows:

“**TM\_Coordinate**”, which gives temporal position represented by a single number, is implemented by XML Schema type “decimal”;

“**TM\_OrdinalPosition**”, which carries an association with a **TM\_OrdinalEra**, is implemented by XML Schema type “anyURI”, which follows the pattern used in GML where associations are implemented through references;

“**TM\_CalDate**”, which carries attributes consisting of sequence of numbers for the calendar date, and an era name, is implemented in **gml:CalDate** by a union (choice) of XML Schema types “date”, “gYear”, “gMonth”, whose lexical representations follow ISO 8601, to which an XML attribute “calendarEraName” is added;

“**TM\_ClockTime**”, which carries a sequence of numbers describing an instant that recurs daily, is implemented by XML Schema type “time”, whose lexical representation follows ISO 8601;

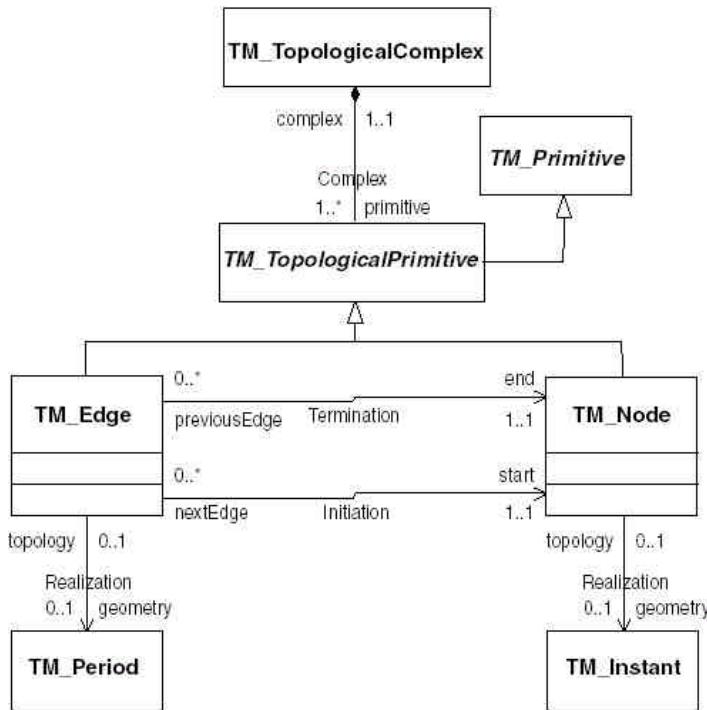
“**TM\_DateAndTime**” is implemented by XML Schema type “dateTime”, whose lexical representation follows ISO 8601;

The variants “**date8601**”, “**time8601**” and “**dateTime8601**”, shown in **TM\_Position**, are implemented by the XML Schema types “date”, “time” and “dateTime”, already introduced;

“**IndeterminatePosition**” is represented using an XML attribute of the same name;

The role “frame” is implemented using an XML attribute of the same name, whose value has type “anyURI”, which follows the pattern used in GML where associations are implemented through references.

#### D.2.4.6 Temporal Topology



**Figure 46** Profile of temporal topology adapted from ISO 19108:2002

“TM\_TopologicalComplex” is represented by the “\_TimeTopologyComplex” object element. The “primitive” role is implemented by a property element of the same name.

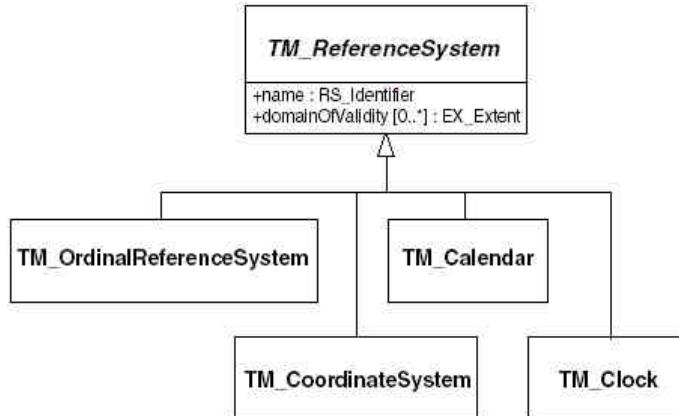
“TM\_TopologicalPrimitive” is represented by the “\_TimeTopologyPrimitive” object element. The “complex” role is implemented as a reference by a property element of the same name, though this is made optional.

“TM\_Node” is implemented by the “TimeNode” object element. The “previousEdge” and “nextEdge” roles are implemented by property elements of the same name in GML. The “geometry” role is implemented by the “position” property.

“TM\_Edge” is implemented by the “TimeEdge” object element. The “start” and “end” roles are implemented by property elements of the same name in GML. The “geometry” role is implemented by the “extent” property.

#### D.2.4.7 Temporal Reference Systems

The following UML class diagrams illustrate the profile of the “Temporal Reference Systems” package (compare with figures 7 to 10 of ISO 19108:2002).

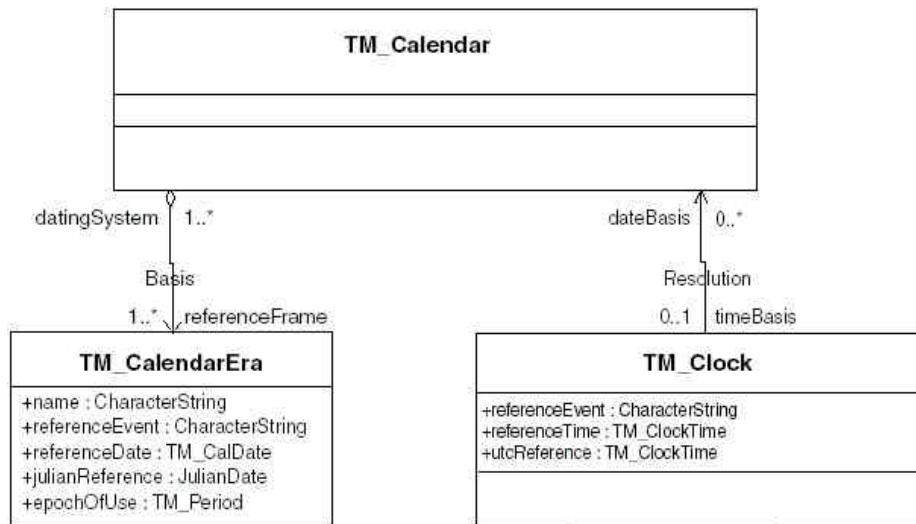


**Figure 47** Hierarchy of temporal reference systems from ISO 19108:2002

“TM\_ReferenceSystem” is implemented by the “\_TimeReferenceSystem” object element. The “\_TimeReferenceSystem” element can carry additional properties: an optional “description” element, one or more “name” elements and an optional “gml:id” attribute. The “domainOfValidity” attribute is implemented using an XML attribute of the same name. This has XML Schema type “string” which implements the “description” attribute of EX\_Extent (see ISO 19115:2003).

These properties are inherited by all the components that are substitutable for \_TimeReferenceSystem.

#### D.2.4.8 Calendars and Clocks



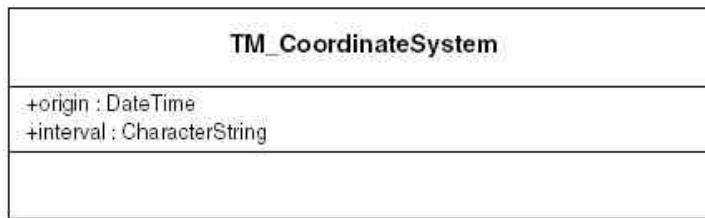
**Figure 48** Model for calendar and clock, from ISO 19108:2002

“TM\_Calendar” is implemented by the “TimeCalendar” object element. The “referenceFrame” role is implemented as a property element of the same name.

“TM\_CalendarEra” is implemented by the “TimeCalendarEra” object element. The “referenceEvent”, “referenceDate”, “julianReference” and “epochOfUse” attributes are implemented as property elements of the same names.

“TM\_Clock” is implemented by the “TimeClock” object element. The “referenceEvent”, “referenceTime”, and “utcReference” attributes are implemented as property elements of the same names. The “dateBasis” role is implemented as a property element of the same name.

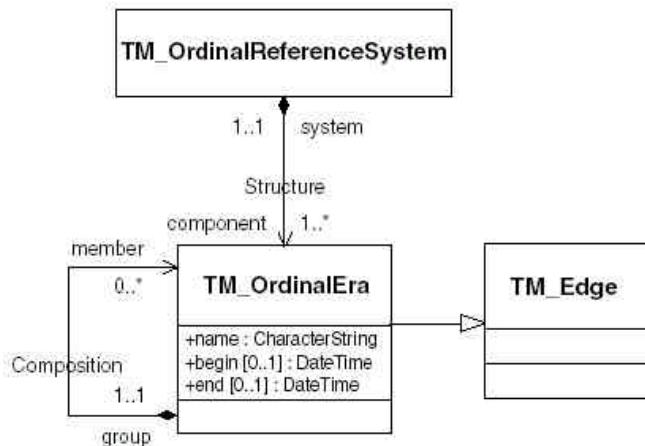
#### D.2.4.9 Time Coordinate Systems



**Figure 49 Model for temporal coordinate system, from ISO 19108:2002**

"TM\_CoordinateSystem" is implemented by the "TimeCoordinateSystem" object element. The "origin" attribute is implemented as a choice of property elements "origin", which refers to a TimeInstant, or "originPosition" which encodes a position directly. The 'interval" attribute is implemented as a property element of the same name, using TimeIntervalLengthType which follows ISO 11404.

#### D.2.4.10 Temporal Ordinal Reference System



**Figure 50 Model for temporal ordinal reference system, adapted from ISO 19108:2002**

"TM\_OrdinalReferenceSystem" is implemented by the "TimeOrdinalReferenceSystem" object element. The "component" role is implemented as a property element of the same name.

"TM\_OrdinalEra" is implemented by the "TimeOrdinalEra" object element. The "name", "begin" and "end" attributes are implemented by the "name", "start" and "end" properties inherited from TimeEdge. The "member" role is implemented as property element of the same name. The "group" role is implemented as a reference by a property element of the same name. Optional "description" and "gml:id" properties are also inherited from TimeEdge.

#### D.2.4.11 Conformance

The rules governing conformance of a profile with ISO 19108 are described in chapter 2 and annex A of ISO 19108. Concerning the criteria defined in chapter 2, GML as an application schema for data transfer targets conformance with A.1. The conditions of this conformance clause are met by the profile specified above.

### D.2.5 ISO 19109 Rules for App Schema

GML implements a subset of the general feature model defined in ISO DIS 19109.

In addition extensions are implemented by GML. The general feature model is concerned only with feature types whereas an application schema (in GML or UML) will often deal with additional information types. Examples are data types, enumeration types, union types, etc. which all cannot be represented using the general feature model. ISO DIS 19109 therefore specifies that only a one-way mapping from the general feature model to the application schema is possible.

Like in the case of UML, the mapping from the general feature model to the GML feature model described in XML Schema is in general straightforward.

The following changes have been applied to the general feature model of ISO 19109.

**Table 13**

	<b>Change</b>	<b>Explanation</b>
1	GF_FeatureOperation deleted	Operation are not supported
2	Multiplicity of GF_InheritanceRelation/supertype changed to "1"	Only single inheritance is supported and mapped to the type derivation mechanism of XML Schema

Some additional comments on the following metaclasses of the general feature model:

**Table 14**

	<b>Change</b>	<b>Explanation</b>
1	GF_AssocationType	<p>In general, the composition of association roles to associations is not directly represented in GML application schemas.</p> <p>If the relationship between two association roles shall be expressed explicitly in the GML application schema, then the roles may be cross-referenced by</p> <ul style="list-style-type: none"> <li>- representing the element name of the target object class in an applInfo annotation with the source "urn:x-gml:targetObjectType"</li> <li>- representing the property name of the inverse association roles in an applInfo annotation with the source "urn:x-gml:inverseProperty"</li> <li>- optionally representing the name of the association in an applInfo annotation with the source "urn:x-gml:associationName"</li> </ul>
2	GF_Constraint	Constraints may be mapped to schematron constraints or may be just expressed as text in documentation annotations.

The relationship between an application schema in UML and in GML is described as part of the annexes E and F dealing with the mapping between ISO 19109 and GML Application Schemas.

## D.2.6 ISO 19111 Spatial Referencing By Coordinates

ISO DIS 19118 Annex A has been used to map the conceptual model to the corresponding XML Schema definitions. This conversion to XML Schemas was done in a manner that makes the results consistent with GML 3.0. Therefore, the encoding rules used were adapted to produce XML Schemas consistent with the GML 3 schema patterns, and to use some of the predefined XML elements and types defined in the GML 3 schemas.

**NOTE** See also the proposed changes to ISO DIS 19118 as a result of the encoding rules for GML application schemas specified in Annex E.

Although the conceptual model is to a very large extent based on ISO 19111, it deviates on a number of points and adds a considerable degree of detail, consistent with the need to provide a specification ready for implementation in the geospatial market place. A list of these variations is:

**NOTE** This list is not completely up-to-date and will need some minor revision. It is the intention that this will result in a proposal for a technical amendment for ISO 19111 (possibly via the WI 19140 process).

- Use of UML

Inheritance of attributes is not dealt with consistently in ISO DIS 19111. In the model used for GML all common attributes and referenced objects have been placed in the highest level (abstract) superclass. This eliminates separate identifiers for SC\_CoordinateReferenceSystem, SC\_CompoundCRS, CC\_Operation and CC\_ConcatenatedOperation.

Conflict with ISO 19111? None. It would not result in any conflict at the implementation level. Only apparent differences exist with the ISO DIS 19111 UML model.

- Coordinate Reference System subtypes

ISO 19111 do not mention subtypes of this class. OGC believes it crucial that this is done, as this subtyping has been part of geodetic practice for many years. The taxonomy is based on the way the effects of earth curvature are dealt with. This has a direct impact on the size of the area for which the coordinate reference system is suited. The main types are:

- geocentric, describing point locations in 3D space; suited to cover the entire earth;
- geographic, describing point locations on or relative to a reference ellipsoid, approximating the geoid over a significant region of the earth;
- projected, treating the earth's geometry as a flat plane, but carefully controlled distortion; typically suited for (parts of) countries;
- engineering, treating the earth's surface as flat, disregarding earth curvature; suited for small areas.

This subtyping has been Implemented by an attribute with enumerated values. This might alternatively be implemented by subclassing, but this would result in a very complex model. A notable addition to the list not mentioned in ISO 19111 is the subtype 'Temporal CRS'.

Conflict with ISO 19111? Yes, although it could be seen as a permitted addition where ISO 19111 describes minimum requirements.

- Coordinate System subtypes

ISO 19111 introduce subtyping at this level, showing however a mixed taxonomy. "Projected" is the outlier here. "Projected" refers to the way earth curvature is dealt with. The ISO taxonomy is, with this exception, by geometry of the coordinate frame. This document honours this subtyping, but adds some types, notably 'linear CS' and 'temporal CS'.

Conflict with ISO 19111? Yes, although it could be seen as a permitted addition where ISO 19111 describes minimum requirements.

- Derived Coordinate Reference Systems

This document adds a concept that is not at all modeled in ISO 19111, viz. 'Derived CRS'. Some CRSs cannot exist without the relationship with a source CRS having been defined, as is the case with a projected CRS. This requires the following changes to the ISO 19111 model:

- a self-reference relationship in SC\_CoordinateReferenceSystem, permitting a source CRS to be defined for a derived CRS;
- a relationship between SC\_CoordinateReferenceSystem and CC\_Conversion, permitting the defining coordinate conversion to be specified.

Making the two relationships from CC\_CoordinateOperation to SC\_CRS optional, to permit the source and target CRS to not be specified in the case of a Defining Conversion. Leaving these relationships mandatory would lead to a potential data conflict, as the information about source and target CRS would be duplicated.

The ISO 19111 URL schema of figure B.1 in ISO 19111 documents a relationship between SC\_CoordinateSystem and SC\_Ellipsoid that is not described in the text. Since the same diagram shows a relationship to CC\_Operation, equally not found back in the text, it is assumed that this is an (unsatisfactory) attempt to model projected CRSs. It is unsatisfactory because the document gives no indication how this is to be done. Furthermore, any literal implementation would result in a number of practical problems. It is for these reasons that the constructs in this document are proposed as an alternative, improved way to achieve inferred objective.

Conflict with ISO 19111? Yes.

- Coordinate operations

Modelling detail has been added in the area of coordinate operations. A new superclass CC\_CoordinateOperation has been created to capture the shared attributes and relationships of the CC\_Operation and CC\_ConcatenatedOperation classes. In addition three more new classes have been created:

- CC\_OperationMethod
- CC\_OperationParameter
- CC\_ParameterValue

One of the two reasons was normalisation, to avoid repetition of attribute values in instantiated objects.

The more important reason was to create a structure that permits being prescriptive about what operation parameters are used by what method. It permits being prescriptive about the operation methods themselves. This part of the model comes close to the heart of coordinate transformation software. The prevalence of distributed computing necessitates standardisation in the data: a coordinate operation may be requested specifying an operation method that is not recognised by the server software. If it does recognise the method, it may not recognise the operation parameters quoted. Is the parameter called 'scale factor at central meridian', 'scale factor at natural origin', 'false magnification', etc? The modelling construct proposed is derived from the EPSG data model, that has proven to be able to resolve this issue.

Conflict with ISO 19111? Yes, but it is quite possible to interpret the extra classes as additional detail implementing the more general ISO 19111 model.

Additionally, some of the changes have been introduced to slightly better support conversion to XML Schemas.

Wherever a GML object is associated with a Coordinate Reference System, this is implemented by an attribute ("srsName") pointing to a gml:CoordinateReferenceSystem element.

## D.2.7 ISO 19112 Spatial Referencing By Geographic Identifiers

GML does not provide a pre-defined schema for gazetteers. However, it does provide pre-defined properties for spatial references by geographic identifiers:

- The property <gml:LocationString> contains or references a text that describes the location.
- The property <gml:LocationKeyword> is a code usually selected from a controlled list, e.g. a ZIP code.

## D.2.8 ISO 19115 Metadata

GML does not provide an information model for metadata. Instead a mechanism to include or reference metadata is provided for all object elements.

**NOTE** As specified in clause 7.5.8, if metadata following the conceptual model of ISO 19115:2003 shall be encoded in a GML document, the corresponding Implementation Specification specified in ISO WD 19139 shall be used to encode the metadata information.

## D.2.9 ISO 19117 Portrayal

### D.2.9.1 Overview

This section provides a comparison of ISO 19117 and the GML Default Style Specification from GML. Since GML Default Style was not directly derived from ISO 19117 some additional background explanation is required.

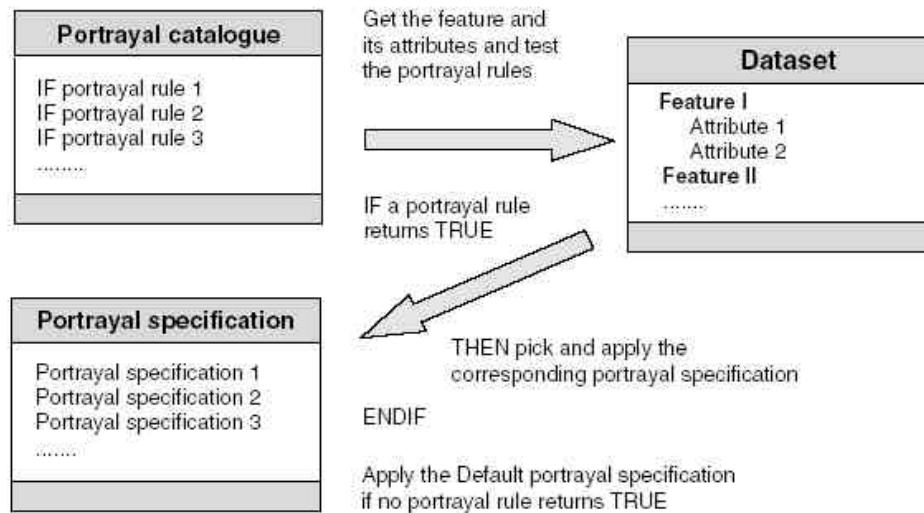
GML Default Style is intended as an implementation specification that provides an XML mechanism for the description of rules for the styling of GML into SVG. It makes use of XML, XML Schema, XLink, XPath (2.0) and SVG. Note that SVG is capable of displaying both vector and image graphics including in particular tiff, png and jpeg image formats.

GML Default Style does not break the principle of separating content and presentation. It does provide a mechanism by which application schema developers can associate graphical presentation styles, expressed in SVG, with GML features. These are to be treated as default styles and may be ignored by a styling application.

This section compares the principles, conformance clauses and specific mechanisms of ISO 19117 with those of this version of GML.

### D.2.9.2 Portrayal Mechanism from ISO 19117

The basic portrayal mechanism from ISO 19117 is illustrated in Figure 51 (this is Figure 2 in ISO/DIS 19117):

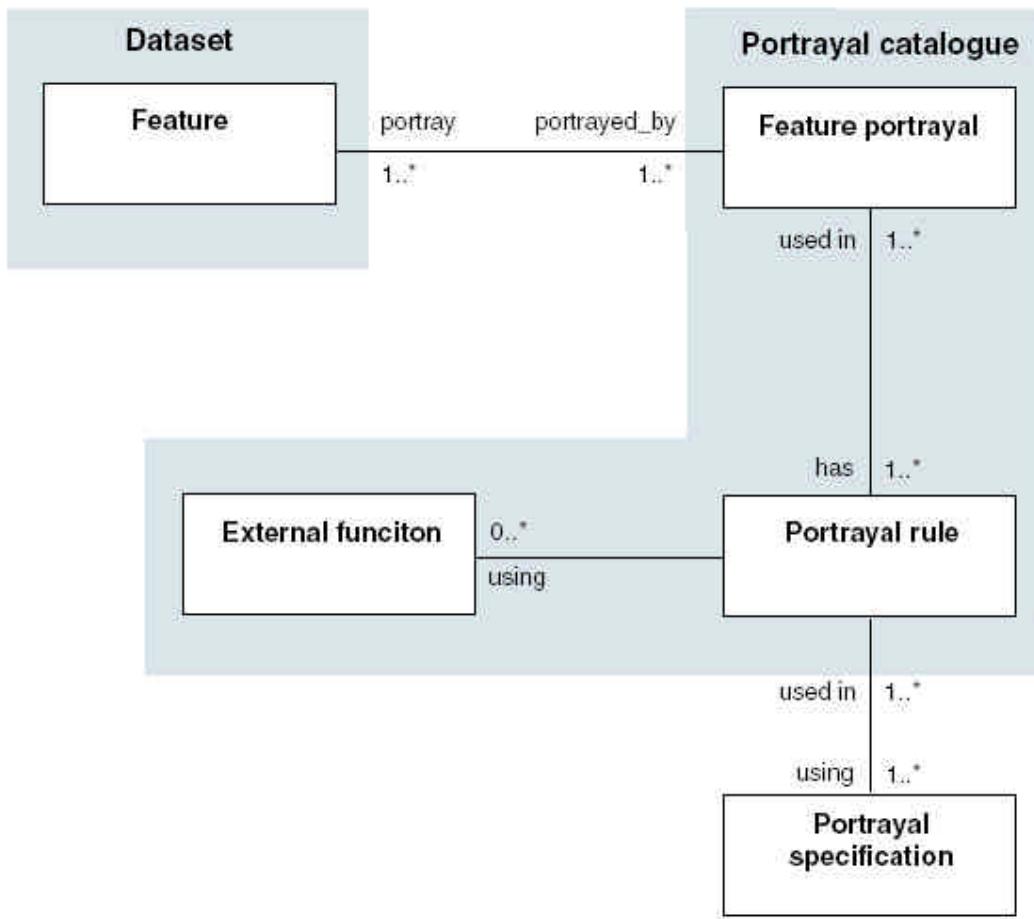


**Figure 51 – ISO 19117 Portrayal Mechanism**

The following basic principles of ISO Portrayal are summarized in the figure:

- It is based on a declarative rule structure (e.g. If (portrayal rule) then Apply (portrayal specification) else (apply default portrayal specification)).
- Datasets and Portrayal Rules are separate from one another or must be capable of being separated from one another.
- Portrayal rules can be organized into catalogues.
- Portrayal rules select what is to be portrayed in the dataset.
- Portrayal specifications state how what is selected in the portrayal rule is to be graphically presented.

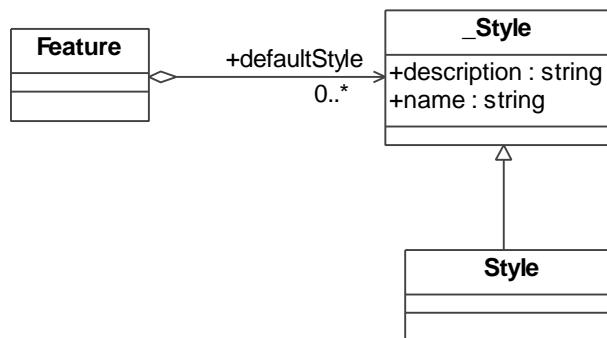
Figure 51 is reinforced by the high level UML model in ISO 19117 shown here as Figure 52.



**Figure 52 – Portrayal UML Diagram from ISO 19117**

Note that Features are associated with 1 or more FeaturePortrayals with the role of the FeaturePortrayal as **portrayed\_by**.

In GML Default Style the property `<gml:defaultStyle>` is introduced. This property has the schema as shown in Figure 53 and is equivalent to ISO 19117 **portrayed\_by** association role.



**Figure 53 – gml:defaultStyle property associates feature instance and style**

The GML defaultStyle property can encapsulate a style definition or can reference one using the xlink:href property. This then corresponds to the UML diagram.

EXAMPLE      <abc:Road gml:id="r1">

```
<gml:defaultStyle xlink:href ="http://www.mysite.org/mystyle.xml" />
<abc:numLanes>3</abc:numLanes>
</abc:Road>
```

Styles in GML default style can be referenced as shown or can appear in-line i.e. inside the gml:defaultStyle property.

GML defaultStyle provides a concrete Style element for which the styling grammar is specified and an abstract substitution group head gml:\_Style to which one can equivalence some other concrete style element with a different styling grammar.

Styles thus correspond to Portrayal Catalogues in ISO 19117. FeaturePortrayal objects consist of the gml:FeatureStyle element together with the featureType attribute.

The Feature Portrayal (ISO 19117) has one or more Portrayal Rules. A Portrayal Rule is in effect a filter to select a feature instance based on its attributes or associations to other features.

EXAMPLE      An example from ISO 19117 document is:

```
(Road.classification .EQ. "country road")
```

In GML Default Style the Portrayal Rule is handled by one of the style elements, namely:

- FeatureStyle (cardinality 0 to unbounded)
- Graph Style

FeatureStyle is intended to describe the portrayal rules and portrayal specification (ISO terminology) for a Feature, and can be correlated to the ISO 19117 PF\_FeaturePortrayal class.

Graph Style has no correlate in ISO 19117 and refers to the portrayal of 1-D graphs from feature collections.

The FeatureStyle element references the associated feature-type using the featureType attribute as in: (this is part of FeaturePortrayal in ISO 19117).

```
<gml:FeatureStyle featureType = "Road"> ... </gml:FeatureStyle>
```

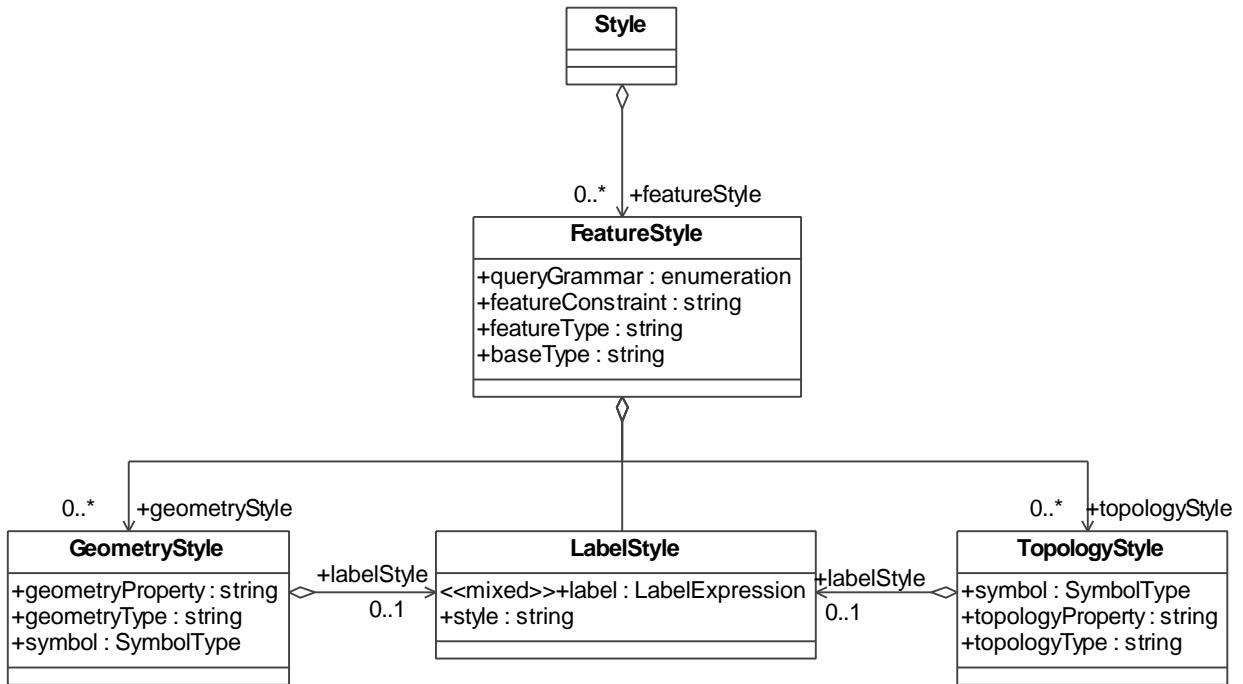
The Portrayal Rule (ISO 19117) is then constructed using the gml:FeatureStyle in conjunction with the featureConstraint. featureConstraint is an XPath expression that is applied against all features of type = feature-type.

EXAMPLE      text(abc:Road/abc:classification) = "country road"

This selects all abc:Road features with classification property of "country road".

The featureConstraint corresponds to the ISO 19117 feature PortrayalRule. The results of applying the XPath expression to a feature of type = feature-type is then styled in accordance with contents of the Feature Style which can include references to external presentation information.

NOTE      GML allows only a single featureConstraint per each FeatureStyle, but that there can be any number of FeatureStyle elements in a Style.



**Figure 54 – Feature Style UML Model**

#### D.2.9.3 Portrayal Service

The Portrayal Service is a service to portray feature instances. It has one method in the ISO UML model, namely `portrayFeature`. Since GML Default Style is only concerned with the implementation of style description such an abstract Portrayal Service is not required in the GML default style specification. The operation `portrayFeature` takes the arguments: a list of feature instances and a portrayalCatalogue consisting of one or more portrayal rules.

#### D.2.9.4 Portrayal Catalogue

This package defines the classes:

- PF\_PortrayalCatalogue
- PF\_FeaturePortrayal
- PF\_PortrayalRule
- PF\_ExternalFunction
- PF\_AttributeDefinition

The contents of the package are shown in Figure 55. Note that a PF\_PortrayalCatalogue is an aggregation of one or more PF\_FeaturePortrayal classes each containing 1 or more PF\_PortrayalRule classes.

Portrayal rules may invoke external functions. Signatures of such external functions are described using PF\_ExternalFunction. Formal parameters associated with these external functions are provided by the PF\_AttributeDefinition class.

The Portrayal Specification is associated with one or more Portrayal Rules.

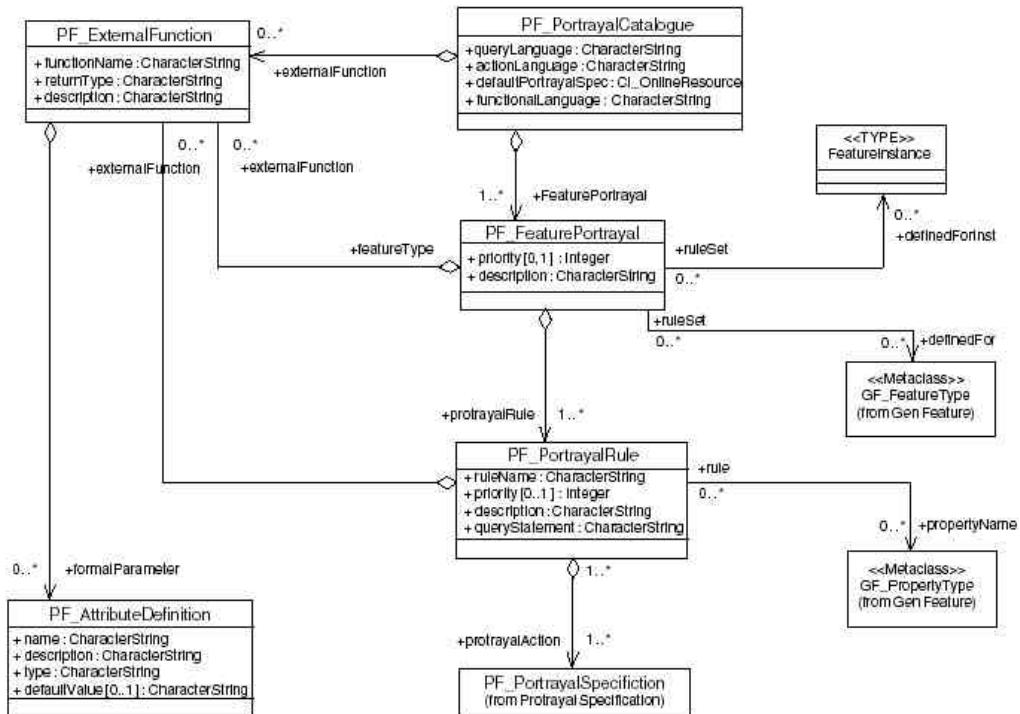


Figure 55 – Portrayal Catalogue UML Model

#### D.2.9.5 PF\_PortrayalCatalogue

This class is an organization of a set of feature portrayal rules. It corresponds to a `gml:Style` in GML default style.

The attributes of **PF\_Portrayal Catalogue** are:

- `queryLanguage` – this specifies notation for how queries are defined.
- `actionLanguage` – this specifies the notion for how actions are performed.
- `defaultPortrayalSpec`
- `functionalLanguage` – name of the functional language used.

In the abstract specification of ISO 19117 `queryLanguage` is OCL and the `actionLanguage` is UML.

In GML Default style the `queryLanguage` is XPath, the `actionLanguage` is XPath/SVG/CSS2 and the `functionalLanguage` is XPath.

The associations of **PF\_PortrayalCatalogue** are:

- `featurePortrayal` – a list of **featurePortrayal** objects
- `externalFunction` – references a list of external functions

In GML default style the `gml:Style` is equivalent to a **PF\_PortrayalCatalogue**.

A `gml:Style` contains one or more `FeatureStyle` elements via `featureStyle` associations. These contain the portrayal rules that are applied to the GML instances.

#### D.2.9.6 PF\_FeaturePortrayal

The `PF_FeaturePortrayal` refers to a feature class through the `featureName` attribute. There is the `definedFor` feature association that refers to the feature type.

NOTE This is what the text says, but not what the UML shows!

In GML default style we have the `featureType` attribute of the `FeatureStyle`. This corresponds to the `PF_FeaturePortrayal` object.

`PF_FeaturePortrayal` has an optional attribute `Priority`. This is not supported in GML default style.

`PF_FeaturePortrayal` has a `description` attribute. In GML default style we have the equivalent `description` attribute on `FeatureStyle` type inherited from the base GML object (`_GML`)

#### D.2.9.7 PF\_PortrayalRule

`PF_PortrayalRule` is a portrayal rule and corresponds in GML to the `featureConstraint` element of `FeatureStyle`. In GML, the content of this element is an XPath/XQuery expression that is assumed to have access to all properties of features referenced by the containing `FeatureStyle` element.

`PF_PortrayalRule` has the following attributes:

- `ruleName` = name of the rule – `gml:name` property of `gml:FeatureStyle` element.
- `priority` = priority between rules – NOT supported in GML.
- `description` = description of the rule – `gml:description` property of `gml:FeatureStyle` element.
- `queryStatement` – a formal specification in the stated query language that evaluates to True or False. In GML this is the `featureConstraint` property. This can be viewed as being applied to the GML instances and evaluating to an XPath node set containing the feature instances that satisfy the constraint OR being used in the `match` attribute of an XSLT template that when true invokes the `action` part of the template. The interpretation is thus implementation dependent.

`PF_PortrayalRule` has the associations:

- (list of) external functions – this is done in GML via the XSLT extension function mechanism.
- `portrayalAction` – formal expression in the action language including pre-processing statements and calls to portrayal operations. This is done in GML through XPath and implicit calls to SVG operations.

#### D.2.9.8 Portrayal Specification

The Portrayal Specification package defines the following classes:

- `PF_PortrayalSpecification`
- `PF_PortrayalOperation`
- `PF_ParameterSet`
- `PF_AttributeValue`

- PF\_AttributeDefinition (Defined in Portrayal Catalogue Package)

#### D.2.9.9 PF\_PortrayalSpecification

This holds instances of the portrayal specification, one for each PF\_PortrayalOperation. The symbol association role points to each related PF\_PortrayalOperation.

The equivalent entity in GML default style is `gml:GeometryStyle` class. It contains all the styling specification for the geometry of the features selected by the featureConstraint filter. GML also defines the `gml:LabelStyle` class which is responsible for specifying the styling information for the text output. The `gml:LabelStyle` can be used as the value of the `gml:labelStyle` property within `gml:GeometryStyle` element.

`gml:GeometryStyle` class has the `gml:symbol` property whose value is an SVG fragment. This value can be referenced remotely or specified inline in the `gml:symbol` property. The value of the symbol specifies all the operations that need to be performed in order to draw the feature in a desired way.

#### D.2.9.10 PF\_PortrayalOperation

`PF_PortrayalOperation` holds the name and description of portrayal operation.

In GML all portrayal operations are SVG drawing operations. SVG does not provide explicit methods, but rather specifies the geometric object and its parameters with the operation part being implicit.

EXAMPLE If we write in SVG

```
<path style="stroke-width:2;fill:none;stroke:brown;" id="" d="M186,479L187,480L190,479L188,477"
      onclick="featureClicked('exp:FillEmbankment:', ')"/>
```

then we can read this as if we have a method `drawPath` with the above parameters.

Thus, `PF_PortrayalOperation` and classes associated with it through `formalParameter` and `parameterSet` associations are all defined implicitly in the SVG.

Explicit function calls in GML Default Style (to map the GML geometry into the SVG geometry) can be done however through an XPath expression that generates the SVG. Note that the ISO 19117 Portrayal Specification does not provide any visibility into the mapping between the geographic objects geometry and the geometry of the graphic portrayal.

#### D.2.9.11 Portrayal of textual data

In GML Default Style, textual data is portrayed using `gml:LabelStyle` element. This element can be used in the `FeatureStyle` element in order to portray one label for the whole feature. It can also be, at the same time, specified inside the `gml:GeometryStyle` (or `gml:TopologyStyle`) element to provide a label for different geometry (or topology) aspect of the feature.

While in ISO 19117 textual data can be handled in two ways: as an attribute of a feature or by annotation schema, the latter option is not available in the GML default style.

#### D.2.9.12 Representation of symbols

GML default style uses symbols via `gml:symbol` property of the `gml:GeometryStyle` element. The property can reference a remote symbol or contain it inline as the value. If referenced remotely, the constraint on reference expression is that it be a valid URI. While it is assumed that the symbols will be managed using symbol libraries and symbol identifiers, it is not a requirement. Any valid URI that evaluates to a symbol can be used.

### D.2.9.13 Tabular Comparison

The UML model of the GML profile defined in this annex describes a conceptual model of the abstract types defined in ISO 19117. The Table 15 maps GML default style class and property names to the corresponding class names and their attributes in ISO 19117 to simplify comparison with that standard.

**Table 15**

GML construct	ISO 19117 construct
gml:_Style (Abstract Class)	Not supported
gml:defaultStyle (property)	portrayed_by (Role)
gml:Style (Class)	PF_PortrayalCatalogue (Class)
gml:featureStyle (property)	featurePortrayal (Association)
gml:FeatureStyle (Class)	PF_FeaturePortrayal (Class)
gml:featureType (attribute)	definedFor (Association)
gml:graphStyle (property)	Not supported.
gml:GraphStyle (Class)	Not supported.
gml:featureConstraint (property)	portrayalRule (Association)
gml:geometryStyle (property)	portrayalAction (Association)
gml:GeometryStyle (Class)	PF_PortrayalSpecification (Class)
gml:topologyStyle (property)	Not Supported
gml:TopologyStyle (Class)	Not Supported
gml:labelStyle (property)	portrayalAction (Association)
gml:LabelStyle (Class)	PF_PortrayalSpecification (Class)
gml:spatialResolution (property)	Not specified
gml:symbol (property)	parameterSet (Association)
SVG Object (the value of gml:symbol property)	PF_ParameterSet (Class)
gml:styleVariation (property)	Handled within PF_ParameterSet (Class)
smil20:animate (property)	Not supported
smil20:animateMotion (property)	Not supported
smil20:animateColor (property)	Not supported
smil20:set (property)	Not supported

### D.2.9.14 Conformance

The following table presents the conformance of the GML Default Styling according to the clauses from the Abstract Test Suite of the ISO Portrayal specification. The clauses were taken from the ISO 19117:2003(E) document.

ISO Abstract Test Suite Clause	GML Conformance
<b>A.1 Portrayal schema</b> <p>a) Test purpose: To verify conformance to the portrayal schema.</p> <p>b) Test method: Verify that the portrayal specifications and portrayal rules are not part of the dataset to be portrayed, and that the portrayal specifications are stored separately and referenced from the portrayal rules. The portrayal specifications may be stored externally and referenced using a universal reference standard like URL. Verify that the portrayal rules are stored in a portrayal catalogue, and that the portrayal rules are specified for the feature classes they will be applied on. Verify that the portrayal rules are expressed using UML.</p>	<b>YES.</b> Portrayal rules are not expressed using UML, except in the document specification of those rules (e.g. XQuery).
<b>A.2 Availability of portrayal information</b> <p>a) Test purpose: Verify that portrayal information is present.</p> <p>b) Test method: Verify that portrayal catalogue and portrayal specification are present, or referenced from appropriate metadata.</p>	<b>YES</b>
<b>A.3 Priority attribute</b> <p>a) Test purpose: To verify correct use of priority attributes.</p> <p>b) Test method: Verify that all the portrayal rules have a priority attribute if priority attributes are used. Verify that if two portrayal rules returning TRUE have the same priority value, then the application shall decide which one takes precedence.</p>	<b>Not Applicable</b> (Priority attribute not used)
<b>A.4 Default portrayal specification</b> <p>a) Test purpose: To verify that a default portrayal specification is present.</p> <p>b) Test method: Verify that a default portrayal specification is associated with the dataset.</p>	<b>YES</b> In GML the style associated with the data set via the <code>gml:defaultStyle</code> property is the default style.
<b>A.5 External function</b> <p>a) Test purpose: To verify correct use of external functions.</p> <p>b) Test method: Verify that the portrayal catalogue lists the external functions used, including the parameters and returned values.</p>	<b>YES.</b> Drawing functions are implicit in the SVG grammar. Other external functions may be obtained from the engine that processes the query language (XQuery), but in general is implementation dependant.

### D.2.10 ISO 19123 Coverages

The UML model of the GML profile defined in this annex describes a conceptual model of the abstract types defined in ISO DIS 19123. Table 16 maps GML Coverage object and property names to the corresponding class names and their attributes in ISO 19123 to ease the comparison with that standard.

**Table 16**

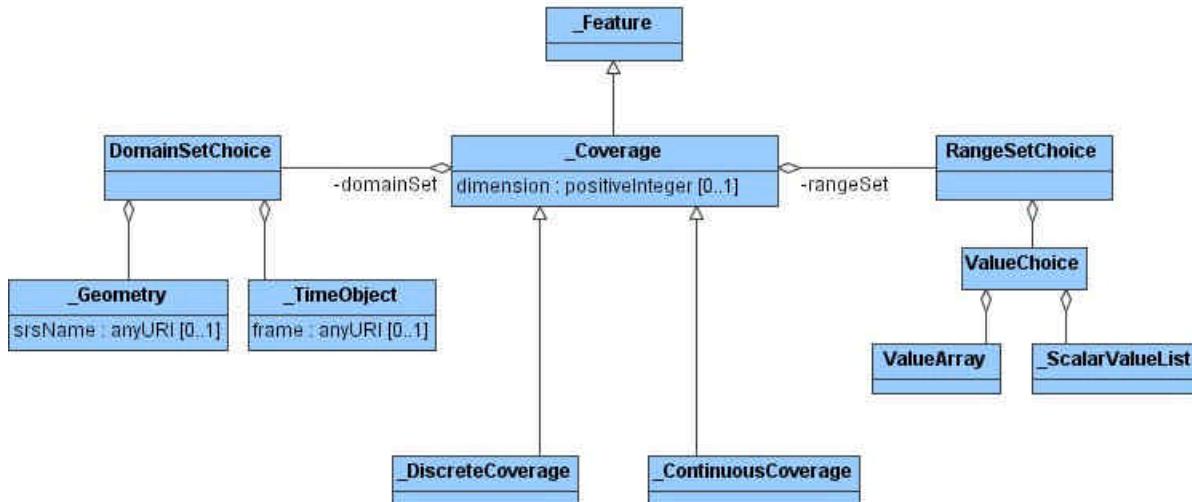
GML construct	ISO 19123 construct
_Coverage (AbstractCoverageType)	CV_Coverage
boundedBy (property)	domainExtent (attribute)
domainSet (property)	domainElement (role name)
rangeSet (property)	rangeElement (role name)
ValueArray, or _ScalarValueList	AttributeValues
coverageFunction (property)	CV_CoverageFunction (Association)
GridFunction	CV_GridValuesMatrix
sequenceRule (property)	sequencingRule (attribute)
SequenceRuleType	CV_SequenceRule
SequenceRuleNames	CV_SequenceType
incrementOrder (attribute)	scanDirection (attribute)
startPoint (property)	startSequence (attribute)
Grid	CV_Grid
GridEnvelope	CV_GridEnvelope
Low (property)	low (attribute)
high (property)	high (attribute)
RectifiedGrid	CV_RectifiedGrid or CV_RectifiableGrid
origin (property)	origin (attribute)
(set of) offsetVector(s) (property)	offsetVectors (attribute)
MultiPointCoverage	CV_DiscretePointCoverage
MultiCurveCoverage	CV_DiscreteCurveCoverage
MultiSurfaceCoverage	CV_DiscreteSurfaceCoverage
GridCoverage or RectifiedGridCoverage	CV_DiscreteGridPointCoverage

The following additional changes have been applied to the coverage package of ISO 19123.

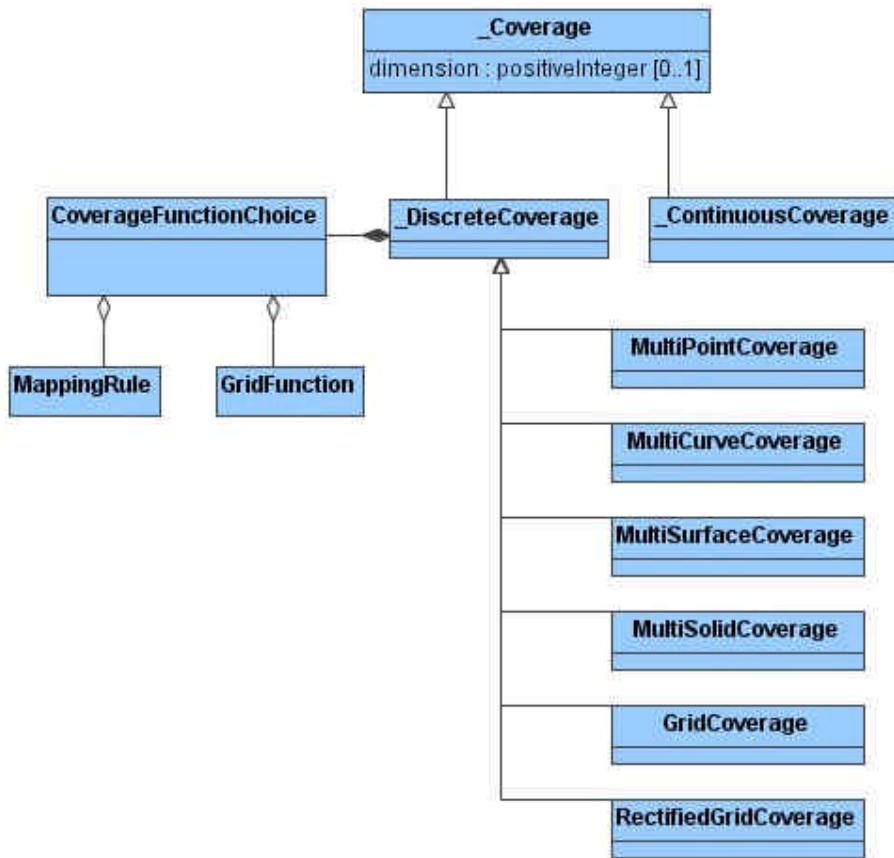
**Table 17**

Change	Rationale
All subclasses of CV_ContinuousCoverage deleted.	Currently not supported by GML
Coordinate Reference System association deleted from CV_Coverage	Replaced by srsName (or frame) attributes on the Geometry or Temporal objects in the domain. The GML coverage package allows for domain objects to be in different coordinate reference systems (or reference frames)
CV_CommonPointRule deleted	Currently not supported by GML
AttributeValues deleted	Replaced by a choice between the GML analogues: ValueArray or _ScalarValueList
CV_GeometrValuePair deleted	Replaced by a choice block consisting of a choice between MappingRule or GridFunction
CV_GridValuesMatrix deleted	The mapping between grid points and range values including sequence rule, etc. is contained in an object (GridFunction) that does not inherit from Grid.
CV_GridCell	Currently not supported by GML

The UML class diagram in Figure 56 illustrates the profile of the “Coverage root” package (compare with Figure 2 of ISO 19123).

**Figure 56**

The UML class diagram in Figure 57 illustrates the discrete coverages (compare with Figures 3, 4 and 5 of ISO 19123).



**Figure 57**

The rules governing conformance of a profile of ISO 19123 are described in chapter 2 and annex A of ISO DIS 19123.

The conditions of the following conformance clauses are met:

- A.1 Simple coverage interface
- A.2 Discrete coverage interface

Note that derived attributes are treated as operations and it is assumed that the derived attributes will be derived from the data by the application handling the GML instances.

## D.3 Extension of the profile of the ISO 19100 Harmonized Model used by GML

### D.3.1 Overview

This section defines the additional parts of GML that are not covered by the profile defined in the previous section. UML is used as the conceptual schema language to describe the additional elements in accordance with ISO PDTS 19103. For details on the semantics of the additional classes see chapters 7 to 20.

For every XML Schema document that is part of the GML namespace a package of the same name (without the ".xsd" extension) is defined. The required additional classes are defined in each package. The packages are part of a package "GML".

### D.3.2 Package "basicTypes"

In addition to those types from ISO PDTS 19103, the schema basicTypes.xsd defines a number of additional types that are used by other GML schemas.

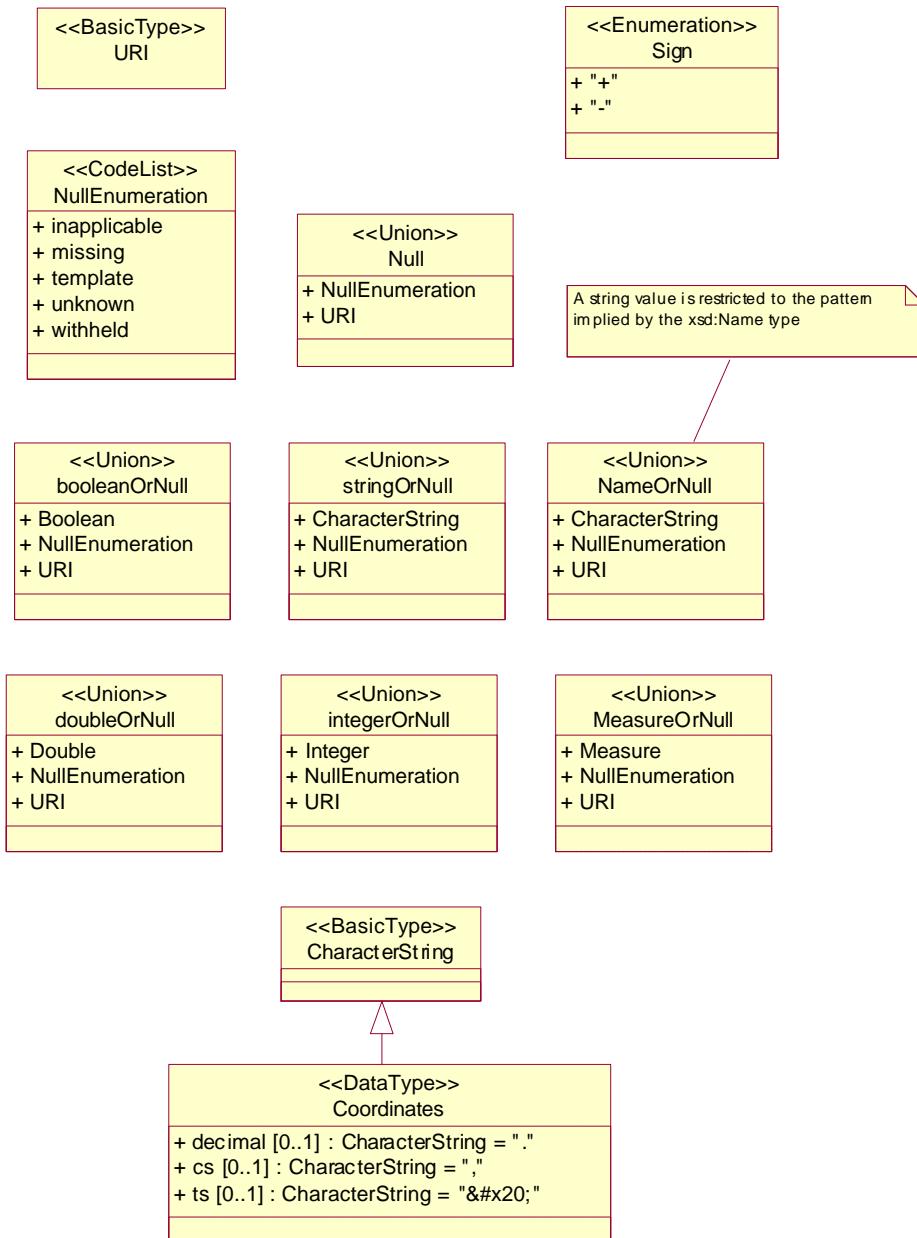
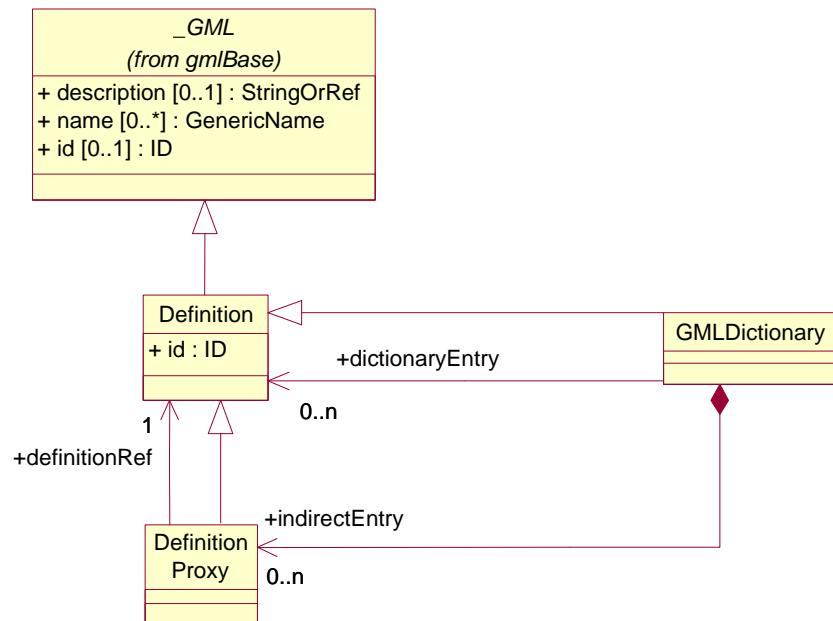


Figure 58

### D.3.3 Package “gmlBase”

The GML Schema gmlBase.xsd does not define additional concepts besides those that are part of the mapping from the conceptual schema to the XML Schema implementation (e.g. representation of object types, associations, etc.; see annex E for more details).

### D.3.4 Package “dictionary”



**Figure 59**

NOTE The “Dictionary” class had to be named GMLDictionary to avoid a naming conflict with the class with the same name in ISO 19103.

### D.3.5 Package “units”

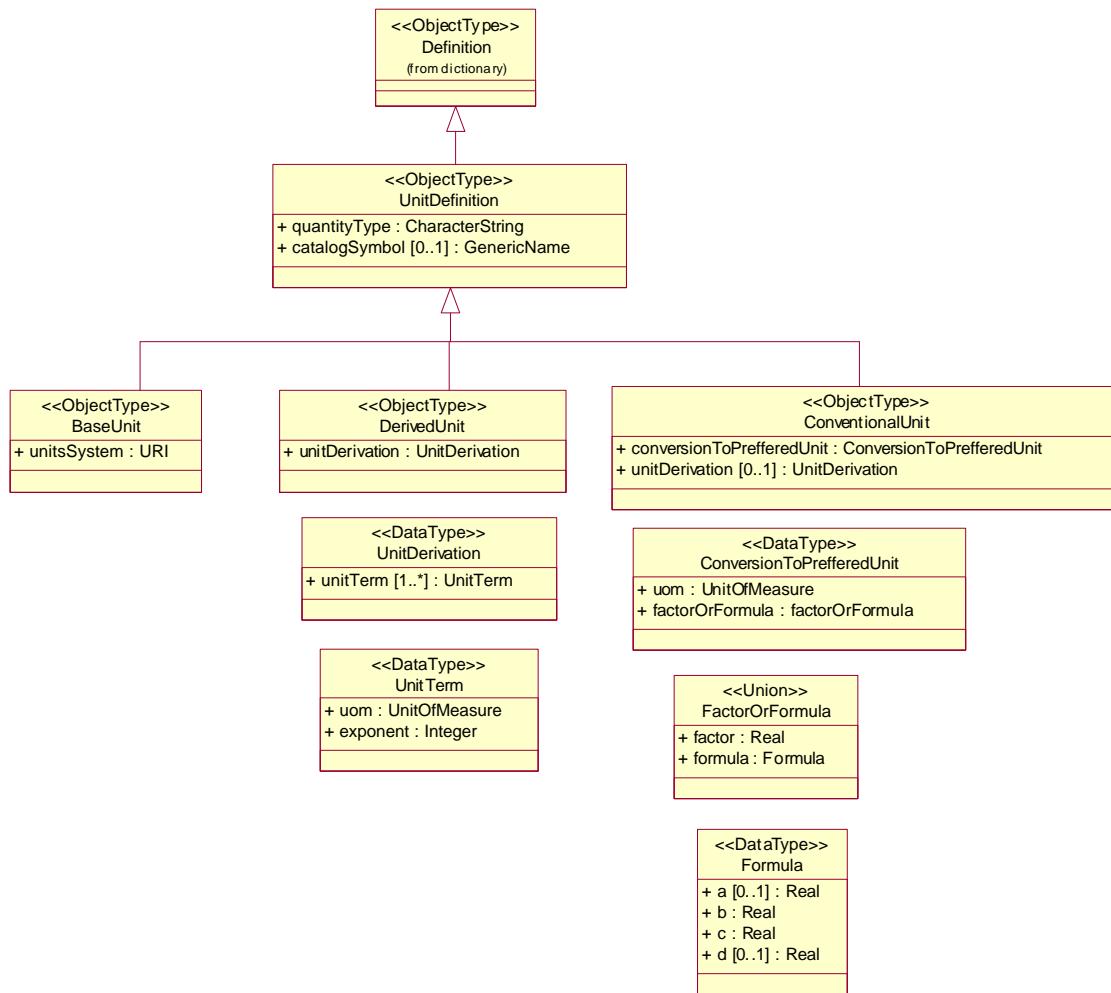


Figure 60

### D.3.6 Package “measures”

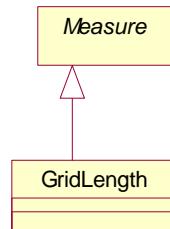


Figure 61

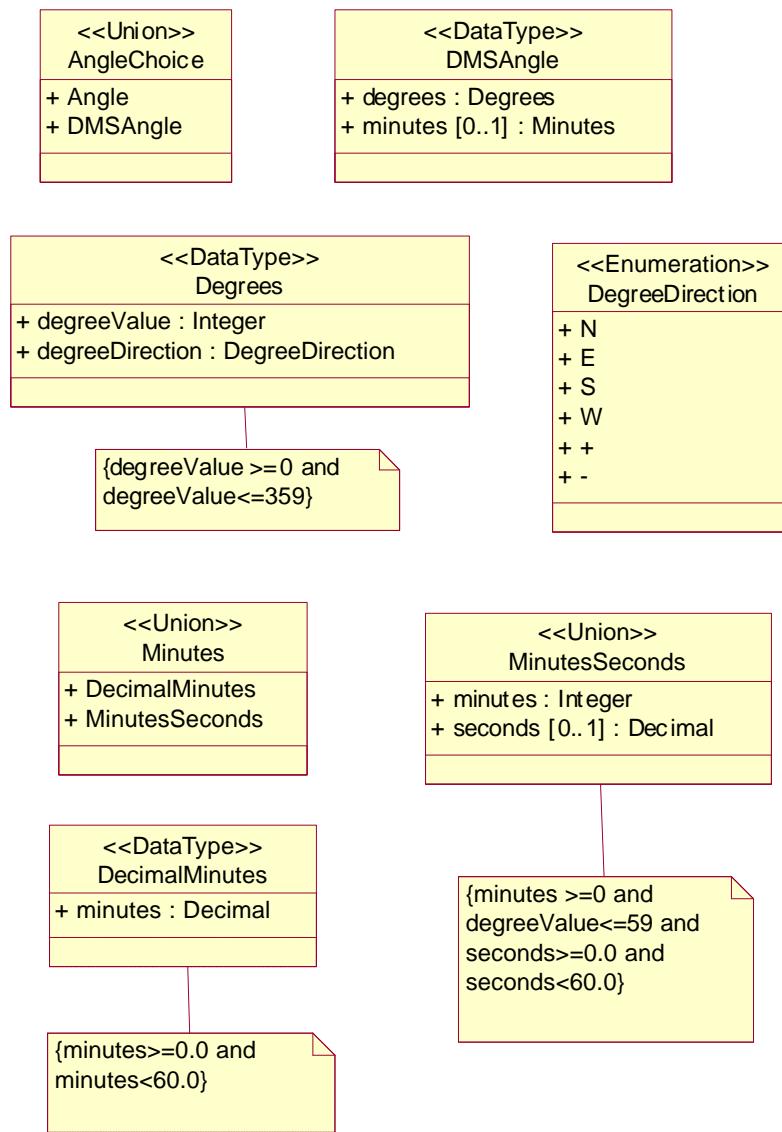
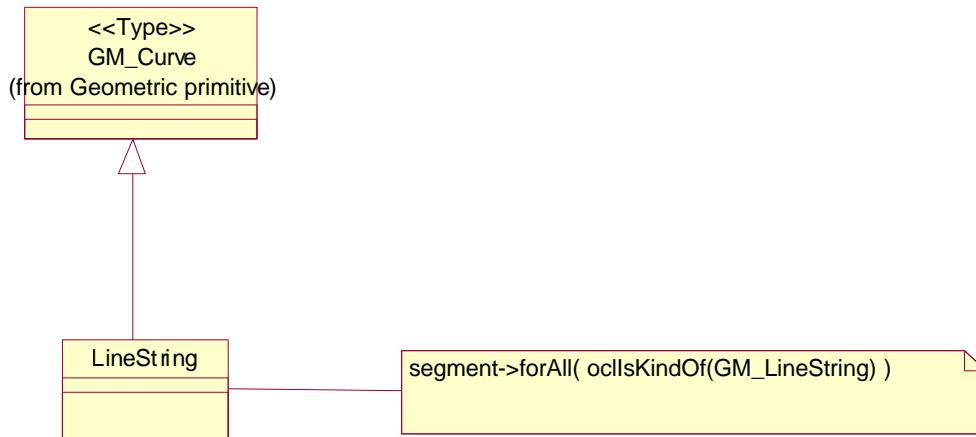


Figure 62

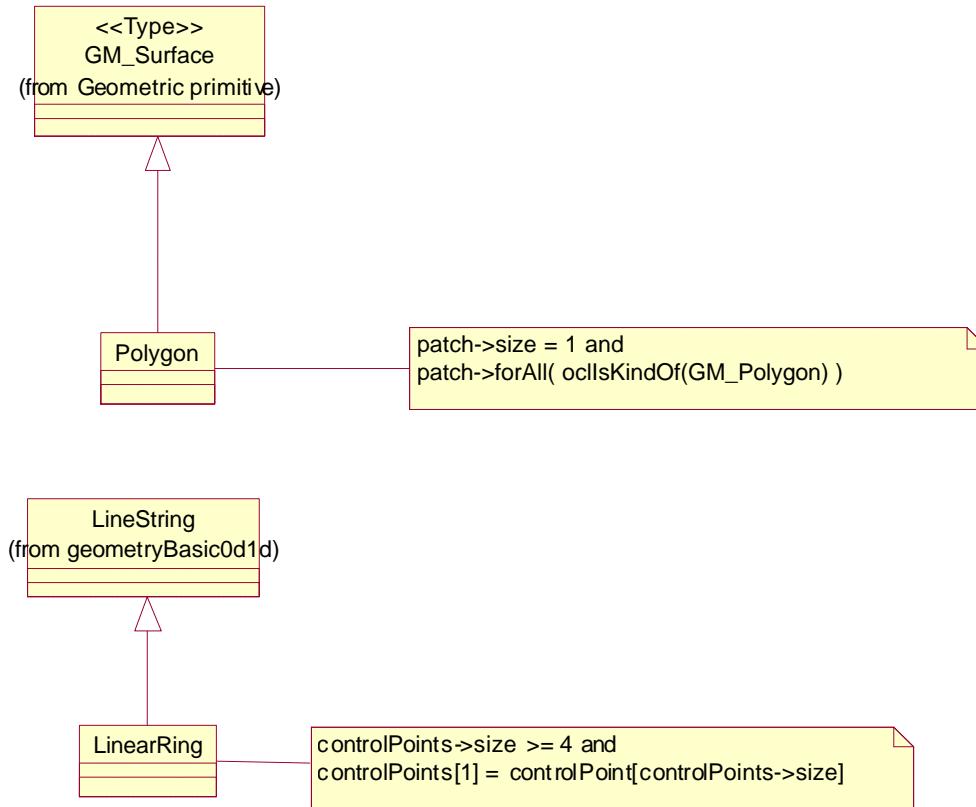
### D.3.7 Package “geometryBasic0d01”



**Figure 63**

An additional subtype of GM\_Curve is added in GML. “LineString” is a special curve that consists of only GM\_LineString segments. The XML representation of the “LineString” object element joins all the segments into one segment and its control points are represented as direct properties of the “LineString”.

### D.3.8 Package “geometryBasic2d”

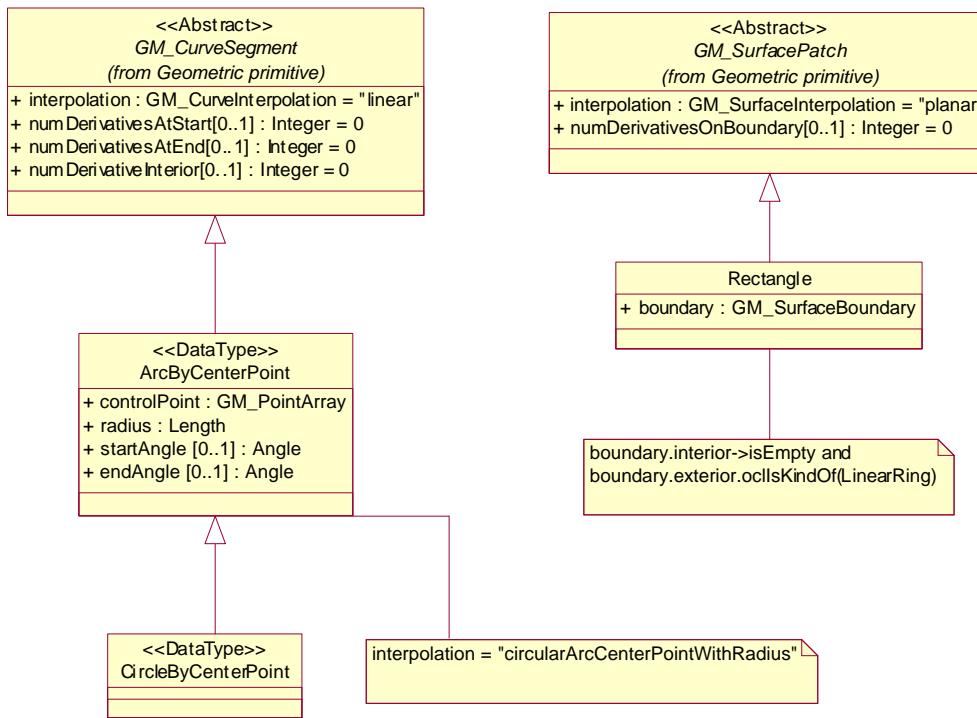


**Figure 64**

An additional subtype of GM\_Surface is added in GML. “Polygon” is a special surface that consists of a single GM\_Polygon patch. Since only a single patch exists, the XML representation of the “Polygon” object element skips the patch level and the exterior and interior boundary properties of the patch are represented as direct properties of the “Polygon”.

In a similar way, “LinearRing” is a simple ring (a GM\_Ring represented by a single line string).

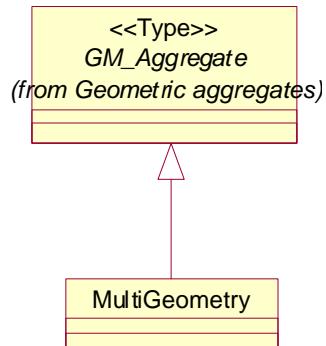
### D.3.9 Package “geometryPrimitives”



**Figure 65**

The curve segments “ArcByCenterPoint” and “CircleByCenterPoint” as well as the surface patch “Rectangle” have been defined in GML.

### D.3.10 Package “geometryAggregates”



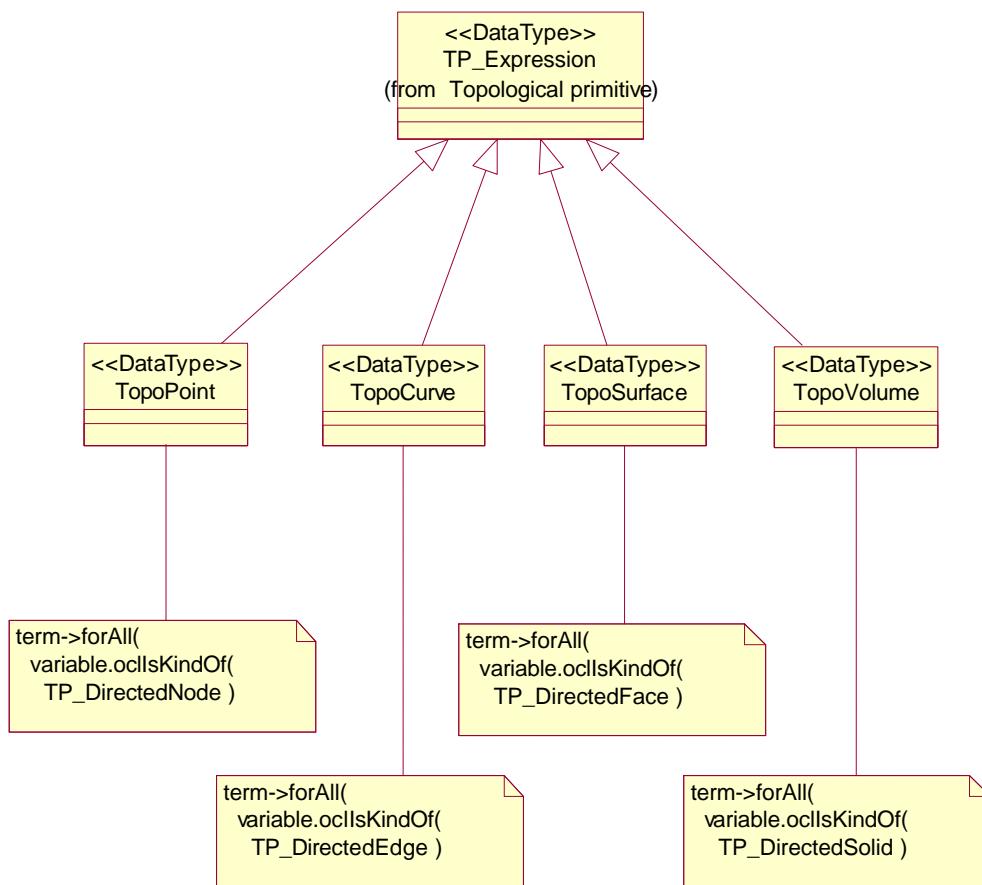
**Figure 66**

GML defines a instantiable geometric aggregate which is not restricted to elements of a single dimension: “*MultiGeometry*”.

### D.3.11 Package “geometryComplexes”

No additional definitions required.

### D.3.12 Package “topology”



**Figure 67**

GML defines several data types containing (or referencing) directed topological primitives, one per dimension: “TopoPoint”, “TopoCurve”, “TopoSurface” and “TopoVolume”. These are intended to be used in properties of features.

### D.3.13 Package “direction”

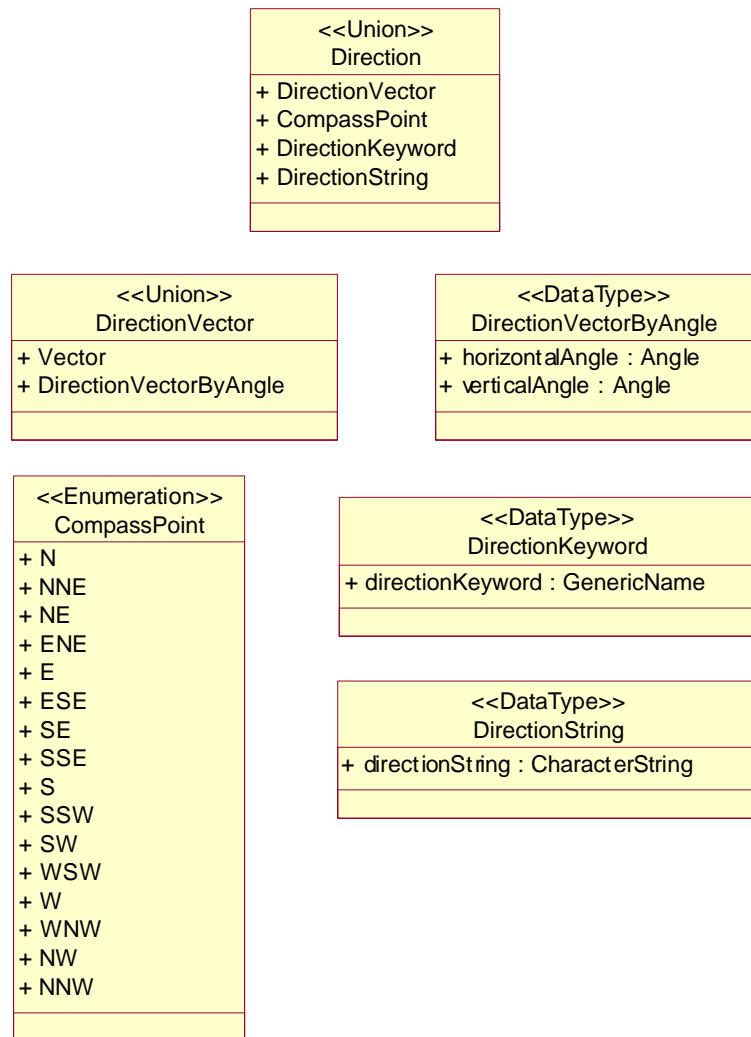


Figure 68

### D.3.14 Package “valueObjects”

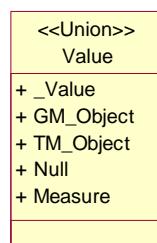
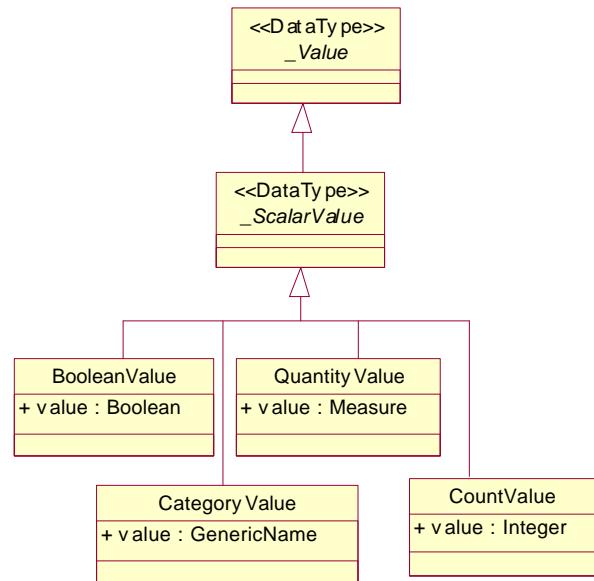
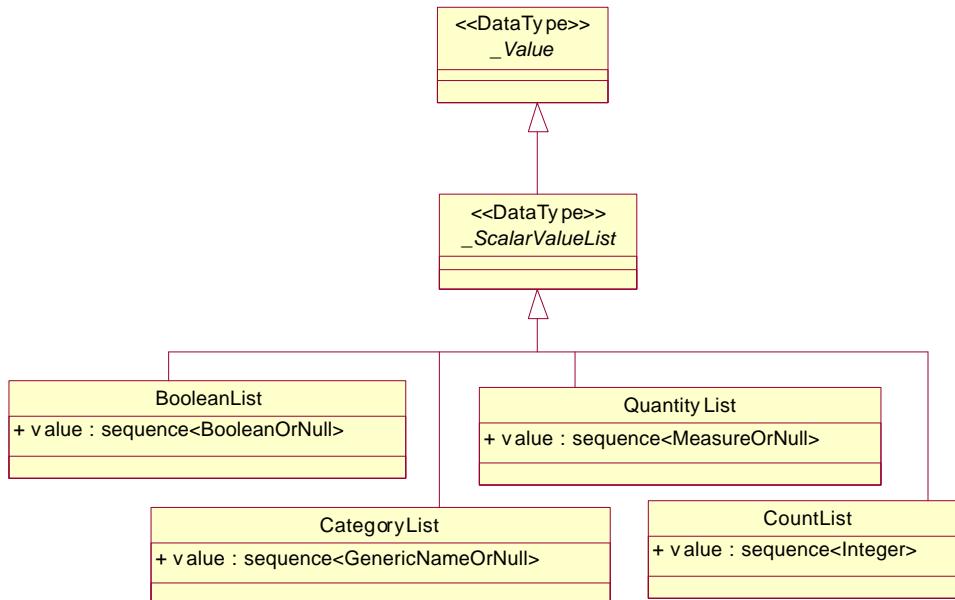


Figure 69



**Figure 70**



**Figure 71**

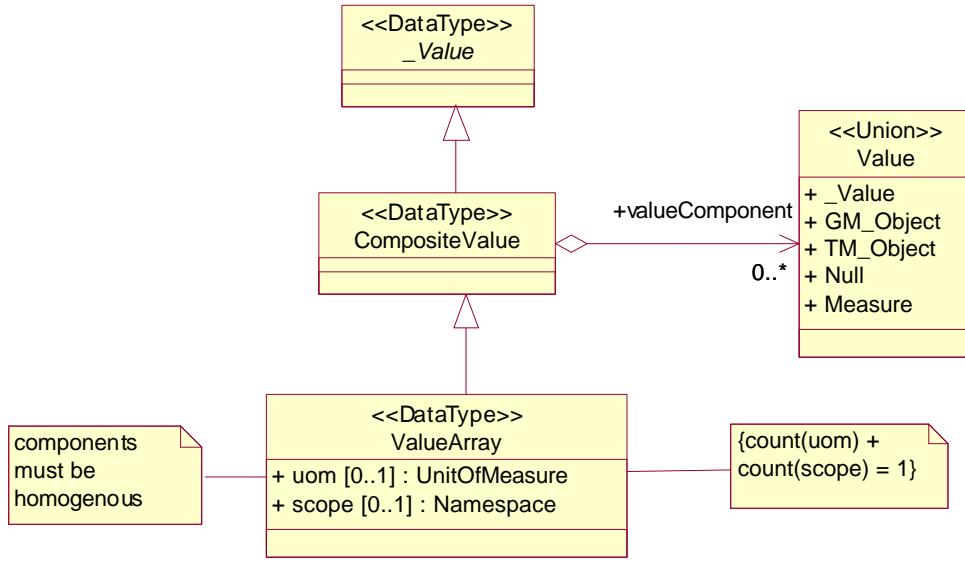


Figure 72

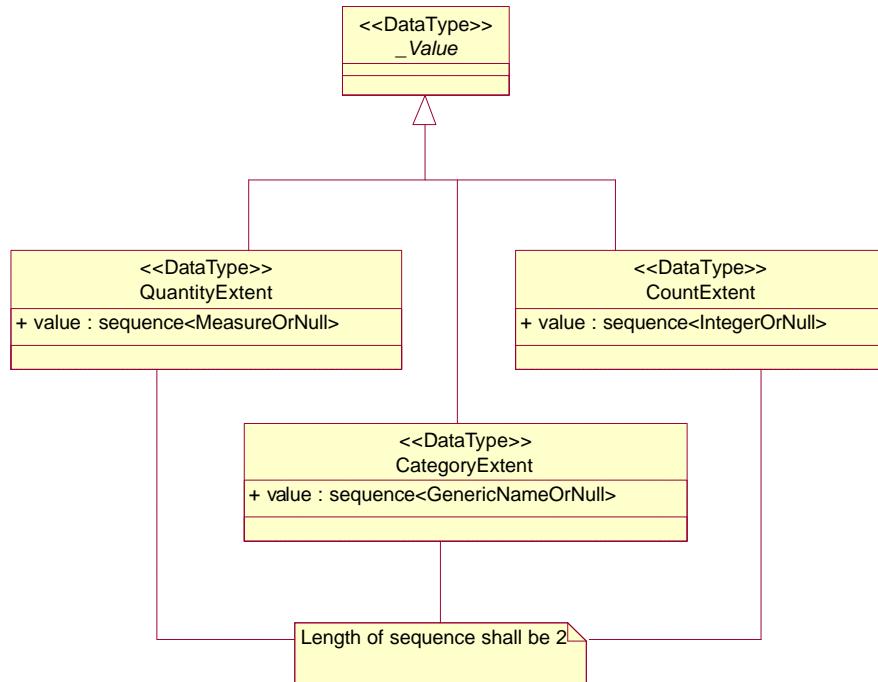


Figure 73

### D.3.15 Package “temporal”

No additional definitions required.

### D.3.16 Package “dynamicFeature”

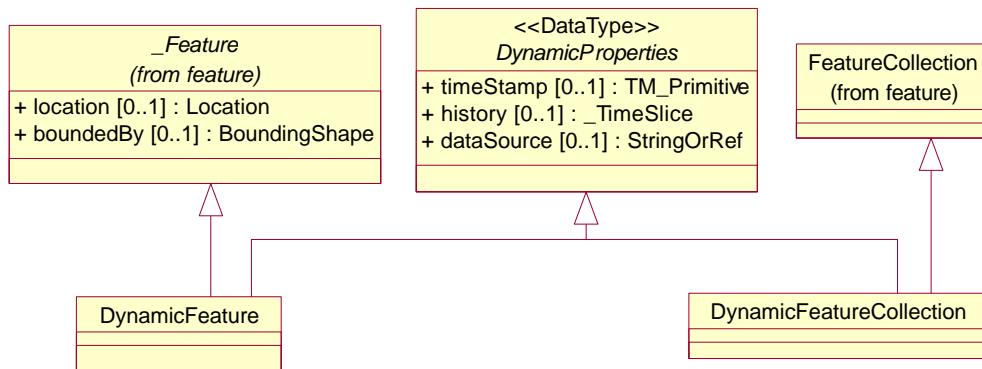


Figure 74

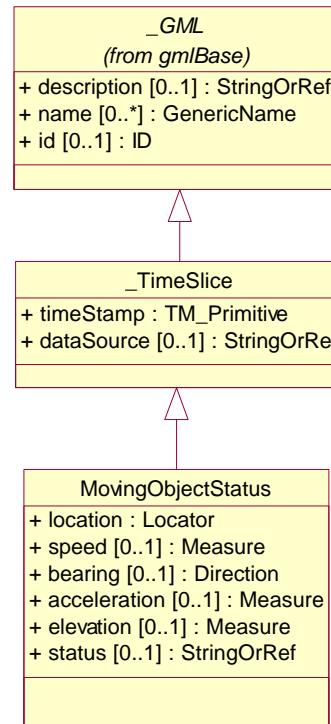
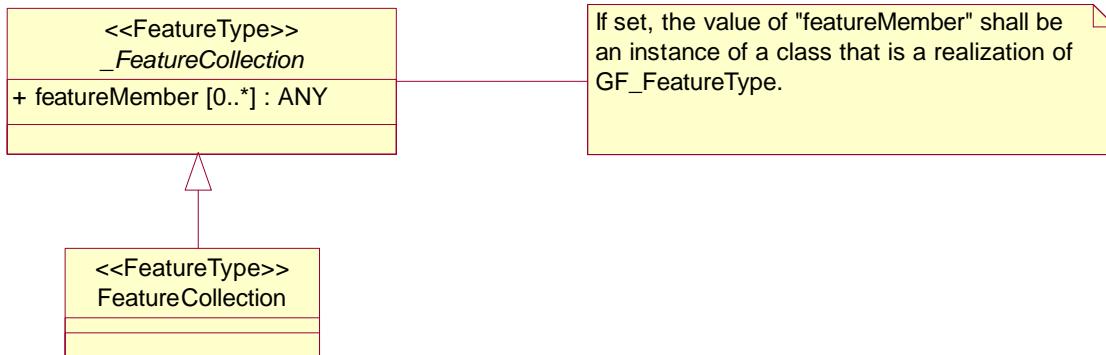


Figure 75

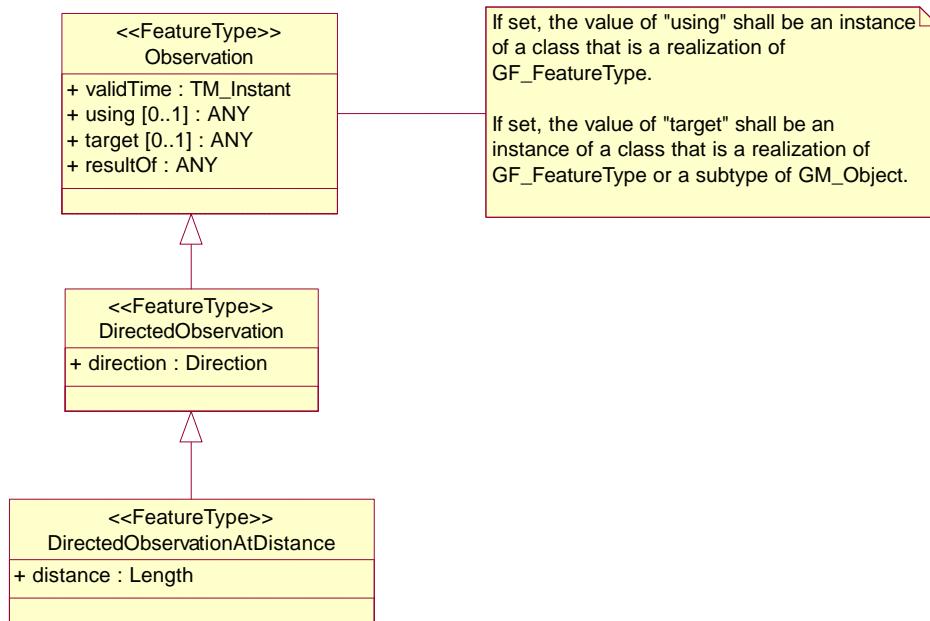
### D.3.17 Package “feature”



**Figure 76**

Beside the definition of the feature collection type, the GML Schema feature.xsd does not define additional concepts besides those that are part of the mapping from the conceptual schema to the XML Schema implementation (representation of feature types and their predefined properties; see annex E for more details).

### D.3.18 Package “observation”



**Figure 77**

### D.3.19 Package “grids”

See clause D.2.10.

**D.3.20 Package “coverage”**

See clause D.2.10.

**D.3.21 Packages “coordinateOperations”, “coordinateReferenceSystems”,  
“coordinateSystems”, “dataQuality”, “datums”, “referenceSystems”**

See clause D.2.6.

**D.3.22 Package “defaultStyle”**

See clause D.2.9.

## Annex E (normative)

### UML-to-GML Application Schema Encoding Rules

#### E.1 General concepts

The mapping from an ISO 19109 conformant UML Application Schema to the corresponding GML Application Schema is based on a set of encoding rules. These encoding rules are compliant with the rules for GML Application Schemas and are based on ISO DIS 19118.

The rules are derived from the current rules for the GML model and syntax as described in clauses 7 to 23, especially clause 7. The (currently informative) encoding rules of ISO DIS 19118 Annex A are used whenever possible and feasible.

The rules listed in this clause (and the subsequent clause E.2) aim at an automatic mapping from an ISO DIS 19109 / ISO PDTs 19103 conformant UML application schema to a GML Application Schema (according to the rules defined in chapter 23). As a result of this automation, the resulting GML Application Schema will not make full use of the capabilities of XML and XML Schema.

These rules do not prescribe that all GML Application Schemas shall be generated by using these rules. All schemas following the rules defined in chapter 23 are valid and conformant GML Application Schemas, whether they are handcrafted, automatically derived from an UML application schema or by some other means.

The schema encoding rules are based on the general idea that the class definitions in the application schema are mapped to type and element declarations in XML Schema, so that the objects in the instance model can be mapped to corresponding element structures in the XML document.

#### E.2 Encoding Rules

##### E.2.1 General encoding requirements

###### E.2.1.1 Application schema

To be a valid input into the mapping the UML Application Schema shall conform to all of the following rules. See ISO DIS 19118 A.2.1 for additional requirements.

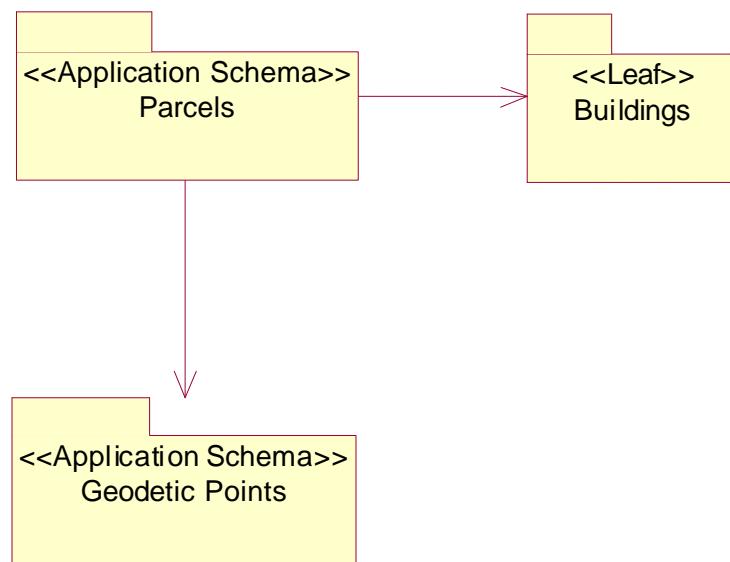
###### E.2.1.1.1 General (Application Schema, Packages)

The UML Application Schema shall conform to the rules defined in ISO DIS 19109 and ISO PDTs 19103.

The UML Application Schema shall be represented by a package with the stereotype <<Application Schema>>. This package shall contain (i.e. own directly or indirectly) all UML model elements to be mapped to object types in the GML Application Schema. The UML model shall be complete and not contain external references unless exceptions are explicitly stated below. This allows that predefined classes may be imported from the standardized schemas of the ISO 19100 Harmonized Model.

All package names shall be unique.

Dependencies between packages shall be modelled explicitly. Permission elements with stereotype <<import>> or unspecified dependency elements between packages shall be used to express the dependency of elements in a package from elements in another package. All other dependency elements shall be ignored.



**Figure 78 – Dependency between packages**

The visibility of all UML elements shall be set correctly. Only publicly visible elements shall be part of Application Schemas used for data interchange between applications.

Documentation of the elements in the UML model shall be stored in tagged values “documentation”.

An XML namespace shall be associated with each package and its sub-packages may be specified by a tagged value (tag name “xmlNamespace” for the URI, tag name “xmlNamespaceAbbreviation” for the abbreviation). These tagged values shall be set for the UML Application Schema package. If they are not set, default values shall be used.

The version of a package, if applicable, shall be specified in a tagged value “version”.

#### E.2.1.1.2 Classes

All class names within the same XML namespace shall be unique.

Feature types shall be modelled as UML classes with stereotype <<FeatureType>><sup>5</sup>.

Object types shall be modelled as UML classes with stereotype <<ObjectType>>. Object types are types where the instances shall have an identity, but which are not feature types<sup>6</sup>.

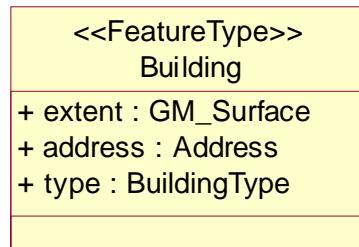
**EXAMPLE** Examples of such types are geometries, topologies, reference systems. Instances of these types can have, for example, a name and an identifier.

<sup>5</sup> Note that it will be easier to identify feature types in UML Application Schemas, if such classes will be tagged with such a stereotype <<FeatureType>>. ISO DIS 19109 currently does not support this stereotype, it is recommended to add such a support in a technical amendment to ISO 19109.

<sup>6</sup> Object types are not considered explicitly in ISO DIS 19109. They appear only as value types of property types.

Abstract object or feature types may have the stereotype <<Abstract>>, but shall be marked as an abstract class (i.e. the UML meta model property “isAbstract” of the class is “true”). The <<Abstract>> stereotype therefore is equivalent to no stereotype.

The same is true for the stereotype <<Leaf>> and classes that are marked as leaf types (i.e. the UML meta model property “isLeaf” of the class is “true”; no subtypes allowed).

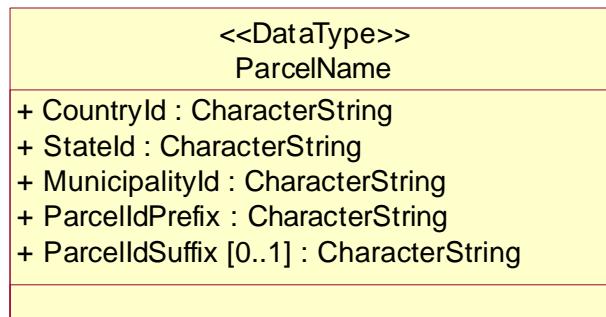


**Figure 79 – A feature type**

Abstract types shall be modelled as UML classes with stereotype <<Interface>> or <<Type>><sup>7</sup>. They may have zero or more operations (these are not mapped to the GML Application Schema), attributes or associations. These classes shall be marked as abstract (the UML meta model property “isAbstract” of the class is “true”).

All subtypes of abstract types shall be either feature, object types or data types.

Data types shall be modelled as UML classes with stereotype <<DataType>>.



**Figure 80 – A data type**

Enumerations shall be modelled as UML classes with stereotype <<Enumeration>>.

Code lists shall be modelled as UML classes with stereotype <<CodeList>>.

---

<sup>7</sup> See ISO PDTS 19103 for a discussion of these stereotypes.

<b>&lt;&lt;CodeList&gt;&gt;</b> <b>ParcelUsage</b>
+ factory = 1 + road = 2 + residential = 3 + offices = 4 + sea, river = 5 + ...

**Figure 81 – A code list**

Union types shall be modelled as UML classes with stereotype <<Union>><sup>8</sup>.

UML classes of the ISO 19100 Harmonized Model that are part of the GML profile of the Harmonized Model may be subclassed in the UML Application Schema. Redefinition of properties is not allowed, only the extension of these classes (i.e. additional properties can be added).

All classes with other stereotypes than those mentioned above may be part of the UML Application Schema, but will be ignored.

Generalization relationships are allowed only between

- feature types and features types,
- object types and object types, or
- data types and data types.

These generalizations shall have no stereotype. All generalization relationships with other stereotypes will be ignored. The discriminator property of the UML generalization shall be blank.

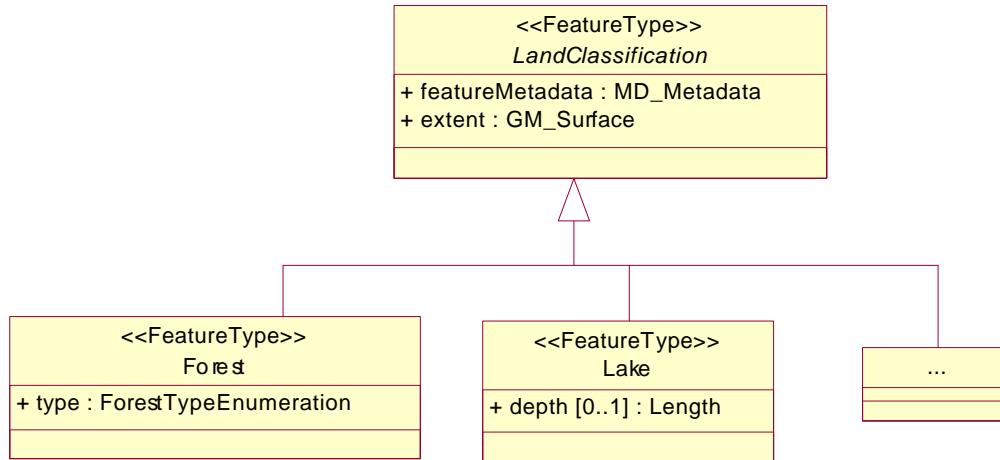
If a class is a specialization of another class, then this class shall have only one supertype (no support for multiple inheritance).

All classes shall have a stereotype specifying the meaning of the class. Classes without a stereotype are treated as feature types<sup>9</sup>.

---

<sup>8</sup> Note that <<Union>> is currently not defined in ISO PDTS 19103 anymore. However, since this stereotype is used quite frequently in ISO 19100 models, it is supported by these encoding rules. The same is true for <<BasicType>>.

<sup>9</sup> If ISO 19109 is amended as proposed above, the default interpretation should be changed to object type.



**Figure 82 – Generalization relationship between feature types**

#### E.2.1.3 Attributes

Every UML attribute of an abstract type, feature type, object type, data type or union type shall have a name and a type. If its multiplicity is not “1”, the multiplicity shall be specified explicitly. An initial value may be specified for attributes with a number, string or enumeration type.

The type shall either be a predefined type (see clause E.2.1.5) or a class defined in the UML model. An attribute with a type that is a feature or object type is treated like an association navigable only towards the association end of the target type.

Every UML attribute of an enumeration class shall have a name. The type information is left empty. No multiplicity, ordering or initial value information shall be attached to the attribute.

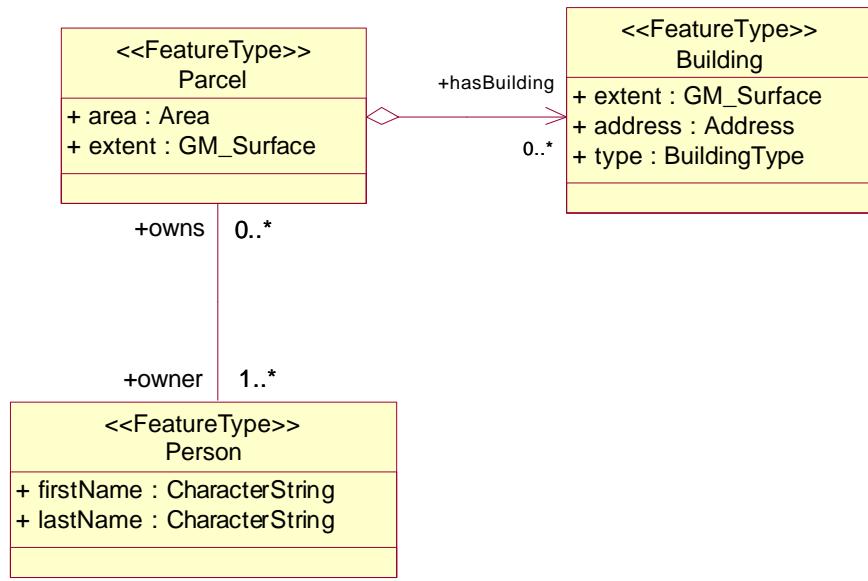
Every UML attribute of a code list class shall have a name. The type information is left empty. No multiplicity or ordering information shall be attached to the attribute. An initial value may be specified to document a code for the code list value. If it is omitted, the value (i.e. the attribute name) is used as the code.

#### E.2.1.4 Associations and association ends

Every UML association shall be an association with exactly two association ends. Both association ends shall connect to a feature or data type and shall have no stereotype or the stereotype <<association>> (or the whole association will be ignored).

The rules for association ends are:

- If an association end is navigable it shall be marked as such and shall have a name. An association end with no name shall be ignored, even if it marked as navigable.
- If the multiplicity of an association end is not “0..\*”, the multiplicity shall be given explicitly.
- The aggregation kind shall be specified explicitly if it is not “none”.
- If the target class of an association end is a data type, then the aggregation kind shall be “composition”.



**Figure 83 – Associations**

#### E.2.1.1.5 Predefined Types

The following predefined types from ISO PDTS 19103 are treated as “basic types” in the sense of ISO DIS 19118 Annex A (i.e. a canonical XML Encoding is attached to them). These basic types are assumed to be pre-defined basic types and shall not be explicitly defined in the UML model:

- Character
- CharacterString
- Integer
- Real
- Decimal
- Number
- Vector
- Date
- Time
- DateTime
- Boolean
- Sign
- Fraction
- GenericName
- LocalName

- ScopedName
- Length
- Distance
- Angle
- Velocity
- Scale
- Area
- Volume
- Measure
- UnitOfMeasure

#### **E.2.1.6 OCL Constraints**

All OCL constraints are ignored. The assessment of the validity of the instance model with respect to these constraints is the task of the application processing the GML instances.

NOTE The Schematron language may be used to express OCL constraints as part of the XML Schema representing the GML Application Schema.

#### **E.2.1.7 Other information**

All other information in the UML Application Schema is not used in the encoding rules and is ignored.

#### **E.2.1.2 Character repertoire and languages**

“UTF-8” shall be used as the character encoding of the XML Schema files (with the associated character repertoire) in accordance with the XML specification.

#### **E.2.1.3 Exchange metadata**

Exchange metadata can be specified for every Feature or Feature Collection in a GML instance document (by using the „gml:metaDataProperty“ element and the ISO 19139 XML Schema implementation specification of ISO 19115). No specific schema for the exchange metadata is added to the GML application schema.

#### **E.2.1.4 Dataset and object identification**

Unique identifiers according to XML's ID mechanism are used to identify objects.

#### **E.2.1.5 Update mechanism**

No explicit update mechanism is defined for the features defined in the GML Application Schema. It is assumed that other mechanisms are used to update a data store.

NOTE An example is the “Transaction” operation of the OpenGIS® Web Feature Service Implementation Specification.

#### **E.2.2 Input data structure**

See ISO DIS 19118 A.3 for a description of the input data structure.

### E.2.3 Output data structure

This encoding rule is based on the XML Recommendation 1.0 and the XML Linking Language (XLink) Version 1.0. The schema for the output data structure that governs the structure of the exchange format shall be a (set of) valid XML Schema(s) according to XML Schema 1.0 and the Rules for Application Schemas (see clause 23).

The XML Schema conversion rules are defined in the following clause.

### E.2.4 Conversion rules

#### E.2.4.1 General concepts

The schema conversion rules define how XML Schema documents (XSDs) shall be derived from an ISO 19109 application schema expressed in UML. A number of general rules are defined in the following clauses to describe the mapping from an UML model that follows the guidelines described in the previous clause.

**NOTE** In this annex the namespace "xsd:" is used to refer to the namespace of XML Schema, which is "<http://www.w3.org/2001/XMLSchema>". The namespace "gml:" refers to the namespace of GML, which is "<http://www.opengis.net/gml>".

The rules are based on the current rules for the GML model and syntax as described in clauses 7 to 23 (especially clauses 7, 8 and 23) and also on the encoding rules of ISO DIS 19118 Annex A.

The schema encoding rules are based on the general idea that the class definitions in the UML application schema are mapped to type and element declarations in XML Schema, so that the objects in the instance model can be mapped to corresponding element structures in the XML document.

The following table gives an overview:

**Table 18 – Schema encoding overview**

Table: UML → GML Application Schema Overview	
UML Application Schema	GML Application Schema
Package	One XML Schema document per package (default mapping)
<<DataType>>	complexType, property type and global element
<<Enumeration>>	Restriction of xsd:string with enumeration values
<<CodeList>>	Union of an enumeration and a pattern (default mapping, an alternative mapping is a reference to a dictionary)
<<Union>>	Choice group whose members are GML Objects or Features, or objects corresponding to DataTypes
<<ObjectType>>	Direct/indirect extension of gml:AbstractGMLType, property type & global element
<<FeatureType>>	Direct/indirect extension of gml:AbstractFeatureType, property type & global element
No stereotype	Direct/indirect extension of gml:AbstractFeatureType, property type & global element
<<Type>>	Direct/indirect extension of gml:AbstractGMLType, property type & global element
Operations	Not encoded
Attribute	local xsd:element, the type is either a property type (if the type is a complex type) or a simple type.

Association role	local xsd:element, the type is always a property type (only named and navigable roles)
General OCL constraints	Not encoded

The multiplicity of attributes and association roles is mapped to „minOccurs“ and „maxOccurs“ attributes in <xsd:element> declarations. The detailed mapping rules are described below.

#### E.2.4.2 UML Packages

The default mapping is that one XML Schema file is generated per package. Name and XML namespace are determined by the name and – if set – the tagged values of the package (see E.2.1). Alternatively, several packages may be mapped to the same XML Schema file. All packages shall be associated with the same XML namespace.

The schema file contains all the XML Schema definitions resulting from the UML classes directly owned by the package(s).

The dependencies of the packages and the package hierarchy are used to determine the required imports and includes of other XML Schema files.

The following tagged values of packages are evaluated:

- `xmlNamespace` and `xmlNamespaceAbbreviation` → URI (e.g. “<http://www.myorg.com/myns>”) and abbreviation (e.g. “myns”) of the target namespace of the XML Schema document.
- `version` → version attribute of the root element of the XML Schema document (default “unknown”).

EXAMPLE      Mapping the information from Figure 78 may result in:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.myorg.com/parcels" xmlns="http://www.w3.org/2001/XMLSchema"
       xmlns:gml="http://www.opengis.net/gml" xmlns:gp="http://www.myorg.com/geodeticPoints"
       xmlns:pcl="http://www.myorg.com/parcels" xmlns:iso19115="http://www.isotc211.org/iso19115/"
       xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDefault="qualified" version="2003-07-20">
    <include schemaLocation="Buildings.xsd"/>
    <import namespace="http://www.myorg.com/geodeticPoints" schemaLocation="GeodeticPoints.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="base/gml3.xsd"/>
    <!-- ... -->
</schema>
```

#### E.2.4.3 UML Classes (general rules)

Allowed stereotypes for UML classes are: no stereotype, <<FeatureType>>, <<ObjectType>>, <<Type>>, <<DataType>>, <<Union>>, <<Abstract>>, <<CodeList>>, <<Enumeration>>, <<Leaf>>, <<Interface>>. All classes will be mapped to the corresponding class category. All UML classes with other stereotypes will be ignored.

All UML classes shall have zero or one supertype.

All UML classes are mapped to named types. A suffix “Type” is added to the name of the type.

#### E.2.4.4 UML Classes (basic types)

The following basic types from the ISO PDTS 19103 profile of GML are predefined and can be referenced in an UML Application Schema. The mapping to a built-in type of XML Schema (“xsd:”) or GML (“gml:”) is specified.

- CharacterString → xsd:string
- Integer → xsd:integer
- Real → xsd:double
- Decimal → xsd:double
- Number → xsd:double
- Vector → gml:VectorType
- Date → xsd:date
- Time → xsd:time
- DateTime → xsd:dateTime
- Boolean → xsd:boolean
- GenericName → gml:CodeType
- LocalName → gml:CodeType
- ScopedName → gml:CodeType
- Length → gml:LengthType
- Distance → gml:LengthType
- Angle → gml:AngleType
- Velocity → gml:SpeedType
- Scale → gml:ScaleType
- Area → gml:AreaType
- Volume → gml:VolumeType
- Measure → gml:MeasureType
- Sign → gml:SignType
- UnitOfMeasure → gml:UnitOfMeasureType

In addition, the following predefined can be referenced in the UML Application Schema and are mapped to a built-in type of XML Schema (“xsd:”) or GML (“gml:”) as follows:

- booleanOrNull → gml:booleanOrNull
- sequence<booleanOrNull> → gml:booleanOrNullList
- stringOrNull → gml:stringOrNull
- sequence<stringOrNull> → gml:stringOrNullList
- NameOrNull → gml:NameOrNull
- sequence<NameOrNull> → gml:NameOrNullList

- doubleOrNull → gml:doubleOrNull
- sequence<DoubleOrNull> → gml:doubleOrNullList
- integerOrNull → gml:integerOrNull
- sequence<integerOrNull> → gml:integerOrNullList
- measureOrNull → gml:MeasureOrNull
- sequence<measureOrNull> → gml:MeasureOrNullList
- Coordinates → gml:CoordinatesType
- Null → gml:NullType

#### E.2.4.4 UML Classes (data types)

UML classes with stereotype <<DataType>> are mapped to XML Schema complex types. If the class has not supertype it is a non-derived type in XML Schema, otherwise it extends its supertype which shall not be derived from gml:AbstractGMLType (directly or indirectly). Abstract superclasses without any attribute or navigable association role are ignored.

Global XML elements with appropriate settings for name (name of the UML class), type (name of the UML class plus “Type”), abstractness (if the class is abstract) and substitution groups (the name of the superclass) may be defined for these classes.

A named complex type shall be created for these classes (carrying the name of the class with a “PropertyType” suffix). The type follows the pattern for association properties as defined in GML (see clause 7.5.3), but without allowing Xlink attributes.

#### EXAMPLE

```
<complexType name="PoleinfoType">
  <sequence>
    <element name="height" type="gml:LengthType"/>
    <element name="measurementDate" type="date"/>
  </sequence>
</complexType>
```

an optionally

```
<element name="Poleinfo" type="ex:PoleinfoType"/>
<complexType name="Poleinfo.PropertyType">
  <sequence>
    <element ref="ex:Poleinfo"/>
  </sequence>
</complexType>
```

or

```
<complexType name="Poleinfo.PropertyType">
  <sequence>
    <element name="Poleinfo" type="ex:PoleinfoType"/>
  </sequence>
</complexType>
```

#### E.2.4.5 UML Classes (feature types)

UML classes with stereotype <<FeatureType>>, without stereotype or one of the stereotypes <<Abstract>>/<<Interface>> derive directly or indirectly from `gml:AbstractFeatureType`. If the class is a class without supertype it extends directly `gml:AbstractFeatureType`, otherwise it extends its supertype which shall be derived from `gml:AbstractFeatureType` (again, directly or indirectly).

There is one special case: A feature type with no supertype and exactly one aggregation or composition to another class may be mapped to a feature collection (i.e. become a restriction of `gml:AbstractFeatureCollectionType`). In this case, the association role shall be defined as substitutable for `gml:featureMember`. The name of the global element would be a combination of the class and the property name to guarantee uniqueness in the XML namespace.

- Global XML elements with appropriate settings for name (name of the UML class), type (name of the UML class plus “Type”), abstractness (true, if the class is abstract) and substitution group (the name of the supertype or “`gml:_Feature`”) are defined for these classes.
- A named complex type may be created for these classes (carrying the name of the class with a “`.PropertyType`” suffix). The type follows the pattern for association properties as defined in GML (see clause 7.5.3).
- A named complex type may be created for these classes (carrying the name of the class with a “`PropertyByValueType`” suffix). The type is a profile of the pattern for association properties as defined in GML restricted to the “by value” form (again, see clause 7.5.3).

EXAMPLE      Mapping “Building” from Figure 79 may be mapped to:

```
<complexType name="BuildingType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element name="extent" type="gml:Surface.PropertyType"/>
        <element name="address" type="pcl:Address.PropertyType"/>
        <element name="type" type="pcl:BuildingTypeType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="Building.PropertyType">
  <sequence>
    <element ref="pcl:Building" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>

<complexType name="Building.PropertyTypeByValueType">
  <sequence>
    <element ref="pcl:Building"/>
  </sequence>
</complexType>

<element name="Building" type="pcl:BuildingType" substitutionGroup="gml:_Feature"/>
```

#### E.2.4.6 UML Classes (object types)

UML classes with stereotype <<ObjectType>> or <<Type>> derive directly or indirectly from `gml:AbstractGMLType`. If the class is a class without supertype it extends directly `gml:AbstractGMLType`, otherwise it extends its supertype which shall be derived from `gml:AbstractGMLType` (again, directly or indirectly), but not from `gml:AbstractFeatureType` (again, directly or indirectly).

- Global XML elements with appropriate settings for name (name of the UML class), type (name of the UML class plus “Type”), abstractness (true, if the class is abstract) and substitution group (the name of the supertype or “`gml_GML`”) are defined for these classes.
- A named complex type may be created for these classes (carrying the name of the class with a “`.PropertyType`” suffix). The type follows the pattern for association properties as defined in GML (see clause 7.5.3).
- A named complex type may be created for these classes (carrying the name of the class with a “`PropertyByValueType`” suffix). The type is a profile of the pattern for association properties as defined in GML restricted to the “by value” form (again, see clause 7.5.3).

#### EXAMPLE

```

<element name="Ellipse" type="ex:EllipseType" substitutionGroup="gml:_CurveSegment"/>

<complexType name="EllipseType">
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <element name="center" type="gml:DirectPosition"/>
        <element name="semiminor" type="gml:VectorType"/>
        <element name="semimajor" type="gml:VectorType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

#### E.2.4.7 UML Classes (enumerations)

UML classes with stereotype `<<Enumeration>>` are mapped to XML Schema simple types. The base type is “`string`”, the domain of values is restricted to the allowed set of literal values as specified by the attribute names of the UML class.

#### EXAMPLE

```

<simpleType name="SignType">
  <restriction base="string">
    <enumeration value="-"/>
    <enumeration value="+"/>
  </restriction>
</simpleType>

```

#### E.2.4.8 UML Classes (code lists)

UML classes with stereotype `<<CodeList>>` are – in principle – mapped like enumerations, but with an added value (`<pattern value='other: \w{2,}'/>`) that allows for any text value beside the predefined values; these free values are prefixed with “`other:`”.

If codes are specified for code list values, either only the code or both the code and the text value shall be represented as enumerated values.

An encoded code value shall be qualified with an `appinfo` annotation (source “`urn:x-gml:codelistValue`”) specifying the text value of the enumerated value.

EXAMPLE      The code list “`ParcelUsage`” from Figure 81 may be represented as:

```

<simpleType name="ParcelUsageType">
  <union memberTypes="pcl:ParcelUsageEnumerationType pcl: ParcelUsageOtherType"/>
</simpleType>

```

```

<simpleType name="ParcelUsageEnumerationType">
  <restriction base="string">
    <enumeration value="1">
      <annotation>
        <appinfo source="urn:x-gml:codelistValue">factory </appinfo>
      </annotation>
    </enumeration>
    <enumeration value="2">
      <annotation>
        <appinfo source="urn:x-gml:codelistValue">road</appinfo>
      </annotation>
    </enumeration>
    <enumeration value="3">
      <annotation>
        <appinfo source="urn:x-gml:codelistValue">residential</appinfo>
      </annotation>
    </enumeration>
    <enumeration value="4">
      <annotation>
        <appinfo source="urn:x-gml:codelistValue">offices</appinfo>
      </annotation>
    </enumeration>
    <enumeration value="5">
      <annotation>
        <appinfo source="urn:x-gml:codelistValue">sea, river</appinfo>
      </annotation>
    </enumeration>
    <enumeration value="factory"/>
    <enumeration value="road"/>
    <enumeration value="residential"/>
    <enumeration value="offices"/>
    <enumeration value="sea, river"/>
  </restriction>
</simpleType>
<simpleType name="ParcelUsageOtherType">
  <restriction base="string">
    <pattern value="other: \w{2,}"/>
  </restriction>
</simpleType>

```

Alternatively, gml:Dictionaries can be used to represent code lists.

#### E.2.4.9 UML Classes (unions)

UML classes with stereotype <<Union>> are mapped as XML Schema complex types. These classes are mapped like data types (see E.2.4.4), but instead of a <xsd:sequence> of the properties, a <xsd:choice> is used so that exactly one of the properties can be specified in an instance of an union.

##### EXAMPLE

```

<complexType name="RemoteResourceType">
  <choice>
    <element name="name" type="string"/>
    <element name="uri" type="anyURI"/>
  </choice>
</complexType>

```

#### E.2.4.10 UML Attributes and Association roles

An UML attribute or association role of an object or feature type is mapped to a local element with the same name in the complex type defining the content model of the object or feature type. The minOccurs and maxOccurs attributes are set according to the definitions in the UML model (see ISO DIS 19118 Annex A for details of the mapping). The type depends on the type of the value of the property in UML:

If the type of the value of the property is of simple content, then the type is used directly.

EXAMPLE    <element name="count" type="integer"/>

If the type of the value of the property is of complex content, then a property type shall be used. The default encoding of the property type allows both the by-value or by-reference representation.

If the encoded property is an association end and the other association end of the association is also encoded in the GML Application Schema, the property name of the other association end shall be encoded in another appinfo annotation (source "urn:x-gml:reverseProperty").

EXAMPLE    By-reference or by-value:

```
<element name="owner" type="ex:Person.PropertyType" minOccurs="0">
  <annotation>
    <appinfo source="urn:x-gml:reverseProperty">owns</appinfo>
  </annotation>
</element>
<complexType name="Person.PropertyType">
  <sequence>
    <element ref="ex:Person" minOccurs="0"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

or

```
<element name="owner" minOccurs="0">
  <annotation>
    <appinfo source="urn:x-gml:reverseProperty">owns</appinfo>
  </annotation>
  <complexType>
    <sequence>
      <element ref="ex:Person" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </complexType>
</element>
```

Alternatively, the property type may support only one of the representations, by-value or by-reference.

EXAMPLE    By-value only:

```
<element name="owner" type="ex:Person.PropertyTypeByValueType" minOccurs="0"/>
<complexType name="Person.PropertyTypeByValueType">
  <sequence>
    <element ref="ex:Person"/>
  </sequence>
</complexType>
```

or

```
<element name="owner" minOccurs="0">
  <complexType>
    <sequence>
      <element ref="ex:Person"/>
    </sequence>
  </complexType>
</element>
```

If only the by-reference representation shall be supported, then the property element shall be qualified with an appinfo annotation (source "urn:x-gml:targetElement") specifying the qualified element name of the target type.

If the encoded property is an association end and the other association end of the association is also encoded in the GML Application Schema, the property name of the other association end shall be encoded in another appinfo annotation (source “urn:x-gml:reverseProperty”).

EXAMPLE By-reference only:

```
<element name="owner" type="gml:ReferenceType" minOccurs="0">
  <annotation>
    <appinfo source="urn:x-gml:targetElement">Person</appinfo>
    <appinfo source="urn:x-gml:reverseProperty">owns</appinfo>
  </annotation>
</element>
```

Depending on the encoding of the class, an UML attribute of a code list or enumeration type is mapped to an element with either a string value (value domain: values of the enumeration or code list) or a URI pointing to the corresponding dictionary entry.

EXAMPLE The code list “BuildingType” may be represented as:

```
<element name="type" type="ex:BuildingTypeType"/>

or

<element name="type" type="gml:ReferencesType">
  <annotation>
    <appinfo source="urn:x-gml:referencedCodelist">BuildingType</appinfo>
  </annotation>
</element>
```

If an UML attribute or UML association is redefined (i.e. a subclass contains an attribute with the same name as in a supertype) then the type derivation takes place in two steps. First a restriction of the superclass is defined by a restriction of the type representing the superclass restricting the redefined attributes and/or specialized associations (the rules for restrictions in XML shall be adhered to by the UML Application Schema). Then this restricted type is extended to include all the additional properties of the subclass. An XML element is generated for both types. The intermediate type and its element will never be instantiated and are both abstract (abstract="true"). The name of the intermediate type shall be the name of the instantiable type where the suffix “Type” is replaced by “RestrictionType”.

#### E.2.4.11 Documentation

Tagged values “documentation” from elements in the UML model are mapped to annotation/documentation elements in the XML Schema files.

EXAMPLE

```
<element name="curveProperty" type="gml:CurvePropertyType">
  <annotation>
    <documentation>This property element either references a curve via the XLink-attributes or contains the curve element. curveProperty is the predefined property which can be used by GML Application Schemas whenever a GML Feature has a property with a value that is substitutable for _Curve.</documentation>
  </annotation>
</element>
```

#### E.2.4.12 Imported Classes from the ISO 19100 Harmonized Model

In addition to the rules defined above, the following rules apply when the UML Application Schema imports classes from the ISO 19100 Harmonized Model.

Classes from the ISO 19100 Harmonized Model that are implemented by the GML Schemas shall be recognized. The use of ISO 19100 classes shall be conformant with ISO DIS 19109. The mapping of the

relevant classes from the ISO 19100 harmonized model according to ISO DIS 19109 is shown in the following table.

**Table 19**

ISO 19100 class	Type of the GML property (attribute or association end)	GML base type (specialized class)	GML substitution element (specialized class)
GM_Object	gml:GeometryPropertyType	gml:AbstractGeometryType	gml:_Geometry
GM_Primitive	gml:GeometricPrimitivePropertyType	gml:AbstractGeometricPrimitiveType	gml:_GeometricPrimitive
GM_Position	gml:geometricPositionGroup (group)	-	-
GM_PointArray	gml:geometricPositionListGroup (group)	-	-
GM_Point	gml:PointPropertyType	gml:PointType	gml:Point
GM_Curve	gml:CurvePropertyType	gml:AbstractCurveType	gml:_Curve
GM_Surface	gml:SurfacePropertyType	gml:AbstractSurfaceType	gml:_Surface
GM_Solid	gml:SolidPropertyType	gml:AbstractSolidType	gml:_Solid
GM_CompositePoint	gml:PointPropertyType	gml:PointType	gml:Point
GM_CompositeCurve	gml:CompositeCurveType	gml:CompositeCurveType	gml:CompositeCurve
GM_CompositeSurface	gml:CompositeSurfaceType	gml:CompositeSurfaceType	gml:CompositeSurface
GM_CompositeSolid	gml:CompositeSolidType	gml:CompositeSolidType	gml:CompositeSolid
GM_Complex	gml:GeometricComplexPropertyType	gml:GeometricComplexType	gml:GeometricComplex
GM_Aggregate	gml:MultiGeometryPropertyType	gml:MultiGeometryType	gml:MultiGeometry
GM_MultiPoint	gml:MultiPointPropertyType	gml:MultiPointType	gml:MultiPoint
GM_MultiCurve	gml:MultiCurvePropertyType	gml:MultiCurveType	gml:MultiCurve
GM_MultiSurface	gml:MultiSurfacePropertyType	gml:MultiSurfaceType	gml:MultiSurface
GM_MultiSolid	gml:MultiSolidPropertyType	gml:MultiSolidType	gml:MultiSolid
GM_MultiPrimitive	gml:MultiGeometryPropertyType	gml:MultiGeometryType	gml:MultiGeometry
TP_Node	gml:DirectedNodePropertyType	gml:DirectedNodeType	gml:DirectedNode
TP_Edge	gml:DirectedEdgePropertyType	gml:DirectedEdgeType	gml:DirectedEdge
TP_Face	gml:DirectedFacePropertyType	gml:DirectedFaceType	gml:DirectedFace
TP_Solid	gml:DirectedTopoSolidPropertyType	gml:DirectedTopoSolidType	gml:DirectedTopoSolid
TP_DirectedNode	gml:DirectedNodePropertyType	gml:DirectedNodeType	gml:DirectedNode
TP_DirectedEdge	gml:DirectedEdgePropertyType	gml:DirectedEdgeType	gml:DirectedEdge
TP_DirectedFace	gml:DirectedFacePropertyType	gml:DirectedFaceType	gml:DirectedFace
TP_DirectedSolid	gml:DirectedTopoSolidPropertyType	gml:DirectedTopoSolidType	gml:DirectedTopoSolid
TP_Complex	gml:TopoComplexMemberType	gml:TopoComplexType	gml:TopoComplex
TM_Instant	gml:TimeInstantPropertyType	gml:TimeInstantType	gml:TimeInstant
TM_Period	gml:TimePeriodPropertyType	gml:TimePeriodType	gml:TimePeriod
TM_TopologicalComplex	gml:TimeTopologyComplexPropertyType	gml:TimeTopologyComplexType	gml:TimeTopologyComplex
TM_Node	gml:TimeNodePropertyType	gml:TimeNodeType	gml:TimeNode
TM_Edge	gml:TimeEdgePropertyType	gml:TimeEdgeType	gml:TimeEdge
SI_LocationInstance	gml:LocationKeyword or gml:LocationString	-	-
types defined in ISO 19115:2003	<i>Property types encapsulating the object and data types from ISO WD 19139 shall be used</i>	-	-

### E.3 Example <informative>

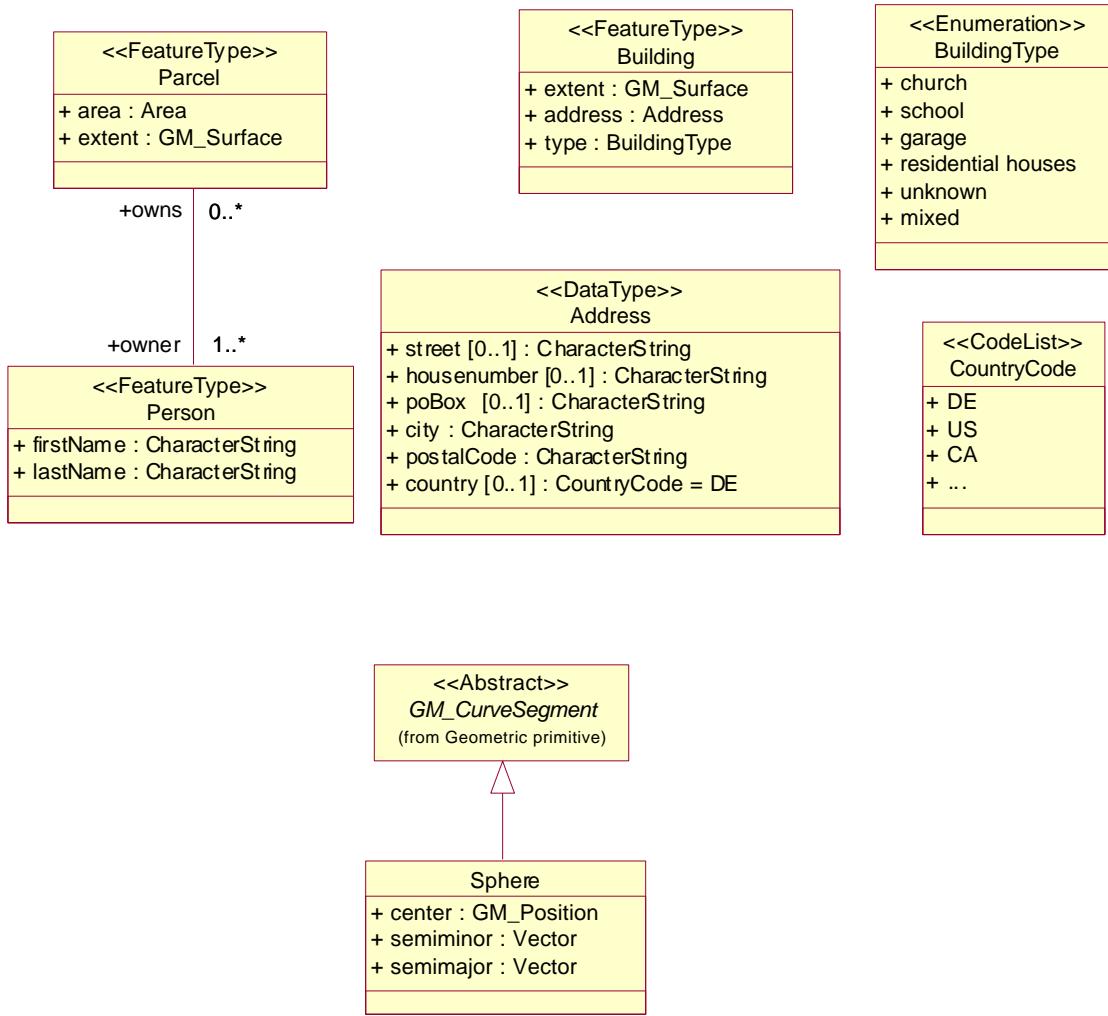


Figure 84 – Example Application Schema

The application schema shown in Figure 84 may be encoded as

```

<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.someorg.de/example" xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:ex="http://www.someorg.de/example" xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified"
  version="1.0">
  <!-- ===== -->
  <import namespace="http://www.opengis.net/gml" schemaLocation=".gmlProfile.xsd"/>
  <import namespace="http://www.w3.org/1999/xlink" schemaLocation=".xlinks.xsd"/>
  <!-- ===== -->
  <element name="Parcel" substitutionGroup="gml:_Feature">
    <complexType>
      <complexContent>
        <extension base="gml:AbstractFeatureType">
          <sequence>
            <element name="area" type="gml:AreaType"/>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
  </element>
  <element name="Building" type="gml:AbstractFeatureType">
    <complexType>
      <complexContent>
        <extension base="gml:AbstractFeatureType">
          <sequence>
            <element name="extent" type="gml:GM_SurfaceType"/>
            <element name="address" type="gml:AddressType"/>
            <element name="type" type="gml:BuildingTypeType"/>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
  </element>
  <complexType name="BuildingType">
    <complexContent>
      <extension base="gml:CodeListType">
        <list value="church" />
        <list value="school" />
        <list value="garage" />
        <list value="residential houses" />
        <list value="unknown" />
        <list value="mixed" />
      </extension>
    </complexContent>
  </complexType>
  <complexType name="Person">
    <complexContent>
      <extension base="gml:AbstractFeatureType">
        <sequence>
          <element name="firstName" type="gml:CharacterStringType"/>
          <element name="lastName" type="gml:CharacterStringType"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="Address">
    <complexContent>
      <extension base="gml:AbstractDataObjectType">
        <sequence>
          <element name="street" type="gml:CharacterStringType"/>
          <element name="housenumber" type="gml:CharacterStringType"/>
          <element name="poBox" type="gml:CharacterStringType"/>
          <element name="city" type="gml:CharacterStringType"/>
          <element name="postalCode" type="gml:CharacterStringType"/>
          <element name="country" type="gml:CountryCodeType" value="DE"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="CountryCode">
    <complexContent>
      <extension base="gml:CodeListType">
        <list value="DE" />
        <list value="US" />
        <list value="CA" />
        <list value="..." />
      </extension>
    </complexContent>
  </complexType>
  <complexType name="GM_CurveSegment">
    <complexContent>
      <extension base="gml:AbstractGeometricPrimitiveType">
        <sequence>
          <element name="semiminor" type="gml:VectorType"/>
          <element name="semimajor" type="gml:VectorType"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
  <complexType name="Sphere">
    <complexContent>
      <extension base="gml:AbstractGeometricPrimitiveType">
        <sequence>
          <element name="center" type="gml:GM_PositionType"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</schema>
  
```

```
<element name="extent" type="gml:SurfacePropertyType"/>
<element name="owner" type="ex:Person.PropertyType">
    <annotation>
        <appinfo source="urn:x-gml:reverseProperty">owns</appinfo>
    </annotation>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<complexType name="Parcel.PropertyType">
    <sequence>
        <element ref="ex:Parcel" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<element name="Building" substitutionGroup="gml:_Feature">
    <complexType>
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element name="extent" type="gml:SurfacePropertyType"/>
                    <element name="address">
                        <complexType>
                            <sequence>
                                <element name="Address" type="ex:AddressType"/>
                            </sequence>
                        </complexType>
                    </element>
                    <element name="type" type="ex:BuildingType"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<complexType name="Building.PropertyType">
    <sequence>
        <element ref="ex:Building" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
<!-- ===== -->
<element name="Person" substitutionGroup="gml:_Feature">
    <complexType>
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element name="firstName" type="string"/>
                    <element name="lastName" type="string"/>
                    <element name="owns" type="ex:Parcel.PropertyType">
                        <annotation>
                            <appinfo source="urn:x-gml:reverseProperty">owner</appinfo>
                        </annotation>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<complexType name="Person.PropertyType">
    <sequence>
        <element ref="ex:Person" minOccurs="0"/>
    </sequence>
    <attributeGroup ref="gml:AssociationAttributeGroup"/>
</complexType>
```

```

<!-- ===== -->
<complexType name="AddressType">
  <sequence>
    <element name="street" type="string" minOccurs="0"/>
    <element name="housenumber" type="string" minOccurs="0"/>
    <element name="poBox" type="string" minOccurs="0"/>
    <element name="city" type="string"/>
    <element name="postalCode" type="string"/>
    <element name="country" type="ex:CountryCode" minOccurs="0" default="DE"/>
  </sequence>
</complexType>
<!-- ===== -->
<simpleType name="BuildingTypeType">
  <restriction base="string">
    <enumeration value="church"/>
    <enumeration value="school"/>
    <enumeration value="garage"/>
    <enumeration value="residential houses"/>
    <enumeration value="unknown"/>
    <enumeration value="mixed"/>
  </restriction>
</simpleType>
<!-- ===== -->
<simpleType name="CountryCodeType">
  <restriction base="string">
    <enumeration value="DE"/>
    <enumeration value="US"/>
    <enumeration value="CA"/>
    <enumeration value="..."/>
  </restriction>
</simpleType>
<!-- ===== -->
<element name="Ellipse" type="ex:EllipseType" substitutionGroup="gml:_CurveSegment"/>
<complexType name="EllipseType">
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <group ref="gml:geometricPositionGroup">
          <element name="semiminor" type="gml:VectorType"/>
          <element name="semimajor" type="gml:VectorType"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</schema>

```

## Annex F (normative)

### GML-to-UML Application Schema Encoding Rules

#### F.1 General concepts

The mapping from a GML application schema to an ISO 19109 conformant UML application schema is based on a set of encoding rules. These encoding rules are compliant with the rules for GML application schemas as described in clauses 7 through 23, especially clauses 7 and 8.

The rules listed in this clause (and the subsequent clause F.2) aim at an automatic mapping from a GML application schema to an ISO DIS 19109 / ISO PDTS 19103 conformant UML application schema.

These rules do not prescribe that all GML Application Schemas shall be generated to fulfill the encoding requirements documented in this annex. All schemas following the rules defined in chapter 23 are valid and conformant GML Application Schemas.

This annex shall be used if there is a requirement in the application domain to derive an ISO 19109 conformant UML Application Schema from a GML Application Schema.

The XML namespace abbreviation "xsd" is used to refer to the namespace of XML Schema, which is "<http://www.w3.org/2001/XMLSchema>".

The XML namespace abbreviation "gml" refers to the XML namespace of GML, which is "<http://www.opengis.net/gml>".

In addition, GML imports definitions from the following namespaces:

The XML namespace abbreviation "xlink" refers to the XML namespace for xlink, which is "<http://www.w3.org/1999/xlink>".

The XML namespace abbreviation "smil20" refers to the XML namespace for SMIL (Synchronized Multimedia Integration Language), which is "<http://www.w3.org/2001/SMIL20>".

The XML namespace abbreviation "smil20lang" refers to the XML namespace for SMIL language, which is "<http://www.w3.org/2001/SMIL20/Language>".

The term "GML namespaces" is used below to refer to the namespaces "gml", "xlink", "smil20", and "smil20lang":

#### F.2 Encoding Rules

##### F.2.1 General encoding requirements

###### F.2.1.1 General remarks

The schema encoding rules are based on the general idea that the corresponding type and element declarations in XML Schema are mapped to class definitions in the UML application schema, so that element structures in the XML document can be mapped to the objects in the instance model.

## F.2.1.2 GML schema

### F.2.1.2.1 General

To be a valid input into the mapping, the GML application schema shall meet the requirements of Annex A.1 Conformance Class B. "Conformance of the XML Implementation of GML Application Schema", and shall also conform to all of the following rules.

The GML application schema shall have and contain definitions for only one target namespace.

The GML application schema may import definitions from XML namespaces other than its target namespace.

A GML application schema consists of a set of one or more XML schema documents such that:

- the documents have unique names;
- the documents contain `xsd:include` elements for other schema documents with the same target namespace;
- one top level schema document for the GML application schema target namespace is not included by any other schema documents for the target namespace, but directly or indirectly includes all other schema documents for the target namespace, if any;
- the schema documents contain `xsd:import` elements for XML namespaces other than the target namespace, and for schema documents that contain definitions in those XML namespaces;
- all included and imported schema documents are accessible via the URI specified by the `schemaLocation` attribute on the `xsd:include` and `xsd:import` elements that reference them;
- a validating XML parser resolves all of the dependencies among the definitions contained in the set of schema documents;
- a validating XML parser validates the set of schema documents without error;
- a validating XML parser validates an XML instance document containing elements and attributes that represent all of the definitions from the target namespace of the GML application schema without error.

Documentation of the definitions contained in a GML application schema shall be stored in nested `xsd:annotation` and `xsd:documentation` elements within the schema definition elements.

The version of a GML application schema, if applicable, shall be contained in the `version` attribute of the `xsd:schema` element from the top level schema for its target namespace.

All global type and element names within a GML application schema shall be unique.

The GML application schema shall not define any elements with anonymous types.

The GML application schema shall not define any XML attributes or named groups.

Every complex type in a GML application schema shall either be a GML object type, GML feature type, a GML data type or a GML property type.

Complex types with simple content shall not be defined in the GML application schema.

The name of all types defined in a GML application schema shall end with the suffix "Type".

A suffix “RestrictionType” in the name of a complex type shall only be used for an abstract type that derives by restriction and which is the base type of exactly one complex type that derives from this type by extension and has the same name as the restricted type except that “RestrictionType” is replaced by “Type”.

A suffix “.PropertyType” in the name of a complex type shall only be used for an instantiable type that follows the pattern for by-reference-or-value property types of GML. A complex type (GML object type or GML feature type) with the same name shall exist that has “.PropertyType” is replaced by “Type”.

A suffix “PropertyByValueType” in the name of a complex type shall only be used for an instantiable type that follows the pattern for by- value property types of GML. A complex type (GML data type, GML object type or GML feature type) with the same name shall exist that has “PropertyByValueType” is replaced by “Type”.

NOTE These rules severely restrict the possible forms of GML application schemas. It is possible that in the future more complex roundtrip mapping rules may be supported.

#### F.2.1.2.2 GML Object Types including GML Feature Types

Each GML object type defined in a GML application schema shall have a content model that directly or indirectly derives from `gml:AbstractGMLType` and shall have a `gml:id` attribute.

Each GML object type of a particular kind defined in a GML application schema shall derive from the most specialized GML object type from the “<http://www.opengis.net/gml>” namespace of a similar kind (with matching semantics) that could possibly be used to define its content model. So GML object types defined in a GML application schema to represent geographic features (GML feature types) shall derive from `gml:AbstractFeatureType` instead of from `gml:AbstractGMLType`, GML object types defined in a GML application schema to represent geometric points shall derive from `gml:PointType` instead of from `gml:AbstractGeometryType`, etc.

GML object types defined in the GML application schema that derive from GML object types outside of the target namespace shall derive directly only from one of the following GML object types from the GML namespace:

- `gml:AbstractGMLType`
- `gml:AbstractGeometryType`
- `gml:PointType`
- `gml:AbstractCurveType`
- `gml:AbstractSurfaceType`
- `gml:AbstractSolidType`
- `gml:CurveType`
- `gml:OrientableCurveType`
- `gml:LineStringType`
- `gml:AbstractCurveSegmentType`
- `gml:SurfaceType`
- `gml:OrientableSurfaceType`
- `gml:PolygonType`
- `gml:AbstractSurfacePatchType`
- `gml:SolidType`

- gml:CompositeCurveType
- gml:CompositeSurfaceType
- gml:CompositeSolidType
- gml :AbstractTopologyType
- gml :NodeType
- gml :EdgeType
- gml :FaceType
- gml :TopoSolidType
- gml :TopoCurveType
- gml :TopoSurfaceType
- gml :TopoVolumeType
- gml :TopoComplexType
- gml:TimeInstantType
- gml:TimePeriodType
- gml:TimeNodeType
- gml:TimeEdgeType
- gml:TimeTopologyComplexType

GML feature types defined in the GML application schema that derive from GML object types outside of the target namespace shall derive directly only from one of the following GML object types from the GML namespace:

- gml:AbstractFeatureType
- gml:AbstractFeatureCollectionType

The schema definitions of abstract GML object types shall contain the attribute “abstract” with the value “true”.

The name of abstract GML object types shall begin with the prefix “Abstract”.

The schema definitions of GML object types for which no subtypes may be defined shall contain the attribute “final” with the value “all”.

The properties of the GML object type shall be specified in a xsd:sequence element.

#### **F.2.1.2.3 Global Elements for GML Object Types**

One global XML element shall be defined for every GML object type defined in a GML application schema.

The name of this element shall be the name of the GML object type without the “Type”-suffix. If the type is abstract, the name prefix “Abstract” shall be replaced by an underscore (“\_”).

The element shall have a substitutionGroup attribute whose value is the name of a global XML element whose type is the base type of the GML object type.

#### F.2.1.2.4 Default property types for GML Object Types

A default GML property type may be defined in a GML application schema for every GML object type defined in that GML application schema.

The GML property type shall either inherit directly or indirectly from `gml:AssociationType` or `gml:ReferenceType`, or it shall be defined according to 7.5.3 “standard pattern for property declarations”.

The name of this property type shall be the name of the GML object type with the “Type”-suffix replaced by “PropertyType”.

If no default property type is specified for a GML object type, an application schema shall use `gml:ReferenceType` as the default property type of the GML object type.

#### F.2.1.2.5 By-value property types for GML Object Types

A default GML property type for by-value properties may be defined in a GML application schema for every GML object type defined in that GML application schema.

The GML property type shall either inherit directly or indirectly from `gml:AssociationType`, or it shall be defined according to 7.5.3 “standard pattern for property declarations”. The use of the `gml:AssociationAttributeGroup` is prohibited in such properties.

The name of this property type shall be the name of the GML object type with the “Type”-suffix replaced by “PropertyByValueType”.

If no default property type for by-value properties is specified for a GML object type, an application schema shall use `gml:AssociationType` as the default property type for by-value properties of the GML object type.

#### F.2.1.2.6 GML Data Types including GML Union Types

A complex type defined in a GML application schema that does not directly or indirectly derive from `gml:AbstractGMLType` is called a GML data type.

The properties of the GML data type shall take one of the following forms:

The content model of the complex type shall not include a `gml:id` attribute.

- The properties of the complex type as well as the properties of all of its base types are specified in a `xsd:sequence` element with `minOccurs` and `maxOccurs` values of “1”.
- The GML data type is not derived from any base type. In this case, the properties may be specified in either a single `xsd:sequence` element with `minOccurs` and `maxOccurs` values of “1” or a single `xsd:choice` element with `minOccurs` and `maxOccurs` values of “1”.

#### F.2.1.2.7 Default property types for GML Data Types

A default GML property type for by-value properties may be defined in a GML application schema for every GML data type defined in that GML application schema.

The GML property type shall either inherit directly or indirectly from `gml:AssociationType`, or it shall be defined according to 7.5.3 “standard pattern for property declarations”. The use of the `gml:AssociationAttributeGroup` is prohibited in such properties.

The name of this property type shall be the name of the GML data type with the “Type”-suffix replaced by “PropertyByValueType”.

If no default property type for by-value properties is specified for a GML data type, an application schema shall use `gml:AssociationType` as the default property type for by-value properties of the GML data type.

#### **F.2.1.2.8    Enumerations**

A simple type defined in a GML application schema that is a restriction of `xsd:string` using only the `xsd:enumeration` facet is called an enumeration.

#### **F.2.1.2.9    Code lists**

A simple type defined in a GML application schema that is a union of an enumeration and a simple type that is a restriction of `xsd:string` using only one `xsd:pattern` facet with the value “other: \w{2,}” is called a code list.

Enumeration values may be qualified with an `applInfo` annotation (source “urn:x-gml:codelistValue”) specifying that the enumeration value is the code value of another enumeration value; the associated enumeration value is given as the text value of the `applInfo` element.

#### **F.2.1.2.10    Global Elements for GML Data Types, enumerations and code lists**

No global XML element shall be defined for GML data types, enumerations or code lists defined in a GML application schema.

#### **F.2.1.2.11    Predefined basic types**

The simple types from the XML Schema and GML namespace listed in the left hand column of Table 22 may be used in the GML application schema. All other simple types from these namespaces shall not be used in a GML application schema.

#### **F.2.1.2.12    GML properties**

Every property of a GML object type (except properties defined in the GML namespace) or of a GML data type shall – with one exception – be represented by a single, locally defined `xsd:element`. Locally defined means that the name and type of the element shall be given explicitly in the element declaration (no references to global XML elements). The element may carry `minOccurs` and `maxOccurs` values. The name of this element shall be the name of the property, the type shall be either a simple type or a property type.

The exception are elements that are substitutable for `gml:featureMember`. These property elements shall be defined as global elements and with a substitution group attribute value of “`gml:featureMember`”. These elements shall only be referenced from GML feature types derived by restriction directly from `gml:AbstractFeatureCollectionType`. The name of such elements shall end with the name of the global element of the target GML feature type.

#### **F.2.1.2.13    Schematron Constraints**

All Schematron constraints are ignored.

#### **F.2.1.2.14    Other information**

All other information in the GML application schema is not used in the encoding rules and is ignored.

#### **F.2.1.3    Character repertoire and languages**

The character encoding used for the schemas determines the available character repertoire.

#### F.2.1.4 Exchange metadata

Exchange metadata can be specified for every Feature or Feature Collection in a GML instance document<sup>10</sup>. No specific schema for the exchange metadata is added to the GML application schema.

#### F.2.1.5 Dataset and object identification

Unique `gml:id` identifiers according to 7.5.6.5 and XML's ID mechanism shall be used to identify GML objects.

#### F.2.1.6 Update mechanism

No explicit update mechanism shall be defined for the feature types defined in a GML application schema. It is assumed that other mechanisms are used to update an instance model data store.

#### F.2.1.7 Input data structure

The schema for the input data structure is defined by the XML Schema 1.0 Part 1: Structures and Part2: Datatypes W3C Recommendations, and the Rules for GML Application Schemas (see clause 23).

### F.2.2 Output data structure

See ISO DIS 19118 A.3 for a description of the output data structure.

### F.2.3 Conversion rules

#### F.2.3.1 General concepts

The schema conversion rules defined in the following clauses describe the mapping from a GML application schema that follows the guidelines described in F.2.1 to a UML application schema that conforms to the rules defined in ISO DIS 19109 and ISO PDTS 19103, using the encoding rules of ISO DIS 19118 Annex A and in particular the generic instance model described in section A.3. These rules are also based on the current rules for the GML model and syntax as described in clauses 7 to 23 (especially clause 7).

The schema conversion rules map definitions from a (set of) valid GML application schema documents (XSDs) to a set of UML packages. A top level package with the stereotype `<<Application Schema>>` is created to contain all the other packages in this set. By default, one package is created in this set for each XSD in the GML application schema, including those directly or indirectly imported from XML namespaces other than the target namespace for the GML application schema, except for XSDs for the GML namespaces. The top level package owns directly or indirectly all UML model elements mapped from object types in the GML application schema.

It is allowed to arrange the definitions of GML application schema in a different package structure as long as the top level package keeps its name and stereotype and all the model elements still belong directly or indirectly to this package.

The type and element declarations in the GML application schema are mapped to class definitions in the UML application schema, so that element structures in the GML XML document can be mapped to corresponding objects in the instance model.

The UML model shall contain the normative packages of the ISO 19100 Harmonized Model or a strict profile of this model.

---

<sup>10</sup> By using the “`gml:metaDataProperty` element and, for example, the ISO WD 19139 XML Schema encoding of ISO 19115:2003.

The UML model shall contain the UML package of all other GML application schemas imported by the GML application schema.

The following table gives an overview, full details of the mapping are specified in the subsequent clauses:

**Table 20 – Schema encoding overview**

Table: GML → UML Application Schema Overview	
GML Application Schema	UML Application Schema
GML application schema	Package <>ApplicationSchema>>
GML schema document {name} XSD	Package named {name}
Object and property type and global element for any object type that is a direct or indirect extension of <code>gml:AbstractFeatureType</code>	Class with stereotype <>FeattrueType>>
Object and property type and global element for any object type that is a direct or indirect extension of <code>gml:AbstractGMLType</code> , other than those that extend <code>gml:AbstractFeatureType</code>	Class with stereotype <>ObjectType>>
Data and property type and global element for any object type that is not a direct or indirect extension of <code>AbstractGMLType</code> and whose content model is a sequence of properties	Class with stereotype <>DataType>>
Restriction of <code>xsd:string</code> with enumeration values	Class with stereotype <>Enumeration>>
Union of an enumeration and a pattern	Class with stereotype <>CodeList>>
Data and property type and global element for any object type that is not a direct or indirect extension of <code>AbstractGMLType</code> and whose content model is a choice of properties	Class with stereotype <>Union>>
Local <code>xsd:element</code> of a <code>simpleType</code> or a <code>complexType</code> with <code>simpleContent</code> or a type that does not directly or indirectly inherit from <code>gml:AbstractGMLType</code>	UML Attribute
Local <code>xsd:element</code> of a type that contains <code>gml:AssociationAttributeGroup</code>	UML Association Role
Schematron constraints	Not encoded

The multiplicity of attributes and association roles is derived from the `minOccurs` and `maxOccurs` attributes in local `xsd:element` declarations.

### F.2.3.2 GML Schema Documents

A top level package with the stereotype <<Application Schema>> is created to contain all the other packages generated for the GML application schema.

- The “xmlNamespace” and “xmlNamespaceAbbreviation” tagged values are applied to the <<ApplicationSchema>> package with corresponding values for the target namespace of the GML application schema

EXAMPLE      “<http://www.myorg.com/myns>” and “myns”

- The “version” tagged value is applied to the <<ApplicationSchema>> package with the default value of “1.0”. If the “version” attribute of the xsd:schema element of the top level schema document for the GML application schema exists and contains a non-empty value, its value replaces the default tagged value.

By default, one UML package is generated for each input schema document in the GML application schema, including those directly or indirectly imported from XML namespaces other than the target namespace of the GML application schema – except for XML Schema documents from the GML namespaces. Alternatively, a single XML Schema document may also be split into several UML packages.

The packages are generated in the <<ApplicationSchema>> package for the GML application schema with names that correspond to the names of the input schema documents.

The xsd:include and xsd:import statements in each input schema document are used to determine and set the dependencies of the packages generated in the <<Application Schema>> package.

### F.2.3.3 GML Object Types

Every GML object type shall be mapped to a UML class.

If the object type directly or indirectly derives from *gml:AbstractFeatureType*, the stereotype of the class shall be <<FeatureType>>, otherwise the stereotype shall be <<ObjectType>>.

The name of the class shall be the same as the name of the global element of the GML object type.

The class shall be abstract, if and only if the GML object type is abstract.

If the GML object type is derived from another GML object type, then the class inherits from the corresponding superclass. If the base type is defined in the GML application schema or another imported GML application schema, then the superclass is the class corresponding to this GML object type. If the base type is defined in the GML namespace, then the superclass is determined by Table 21.

**Table 21**

GML object type	ISO 19100 class
<i>gml:AbstractGeometryType</i>	GM_Object
<i>gml:AbstractGeometricPrimitiveType</i>	GM_Primitive
<i>gml:PointType</i>	GM_Point
<i>gml:AbstractCurveType</i>	GM_Curve
<i>gml:AbstractSurfaceType</i>	GM_Surface
<i>gml:AbstractSolidType</i>	GM_Solid
<i>gml:PointType</i>	GM_CompositePoint
<i>gml:CompositeCurveType</i>	GM_CompositeCurve
<i>gml:CompositeSurfaceType</i>	GM_CompositeSurface

gml:CompositeSolidType	GM_CompositeSolid
gml:GeometricComplexType	GM_Complex
gml:MultiGeometryType	GM_Aggregate
gml:MultiPointType	GM_MultiPoint
gml:MultiCurveType	GM_MultiCurve
gml:MultiSurfaceType	GM_MultiSurface
gml:MultiSolidType	GM_MultiSolid
gml:MultiGeometryType	GM_MultiPrimitive
gml:DirectedNodeType	TP_Node
gml:DirectedEdgeType	TP_Edge
gml:DirectedFaceType	TP_Face
gml:DirectedTopoSolidType	TP_Solid
gml:DirectedNodeType	TP_DirectedNode
gml:DirectedEdgeType	TP_DirectedEdge
gml:DirectedFaceType	TP_DirectedFace
gml:DirectedTopoSolidType	TP_DirectedSolid
gml:TopoComplexType	TP_Complex
gml:TimeInstantType	TM_Instant
gml:TimePeriodType	TM_Period
gml:TimeTopologyComplexType	TM_TopologicalComplex
gml:TimeNodeType	TM_Node
gml:TimeEdgeType	TM_Edge

The GML properties of the GML object type shall be mapped to attributes and association roles as described in clause F.2.3.9.

#### F.2.3.4 GML Object Types (imported from the GML schemas)

The complex types from the GML namespace listed in the left hand column of Table 21 shall be mapped to the predefined UML classes from the ISO 19100 profile of GML in the right hand column of the table.

#### F.2.3.5 Basic Types

The simple types from the XML Schema and GML namespace shown in the left hand column of Table 22 shall be mapped to the predefined UML classes from the ISO 19100 profile of GML in the right hand column of the table.

**Table 22**

XML type	UML class
xsd:string	CharacterString
xsd:boolean	Boolean

xsd:float	Real
xsd:double	Real (Default), Decimal or Number
xsd:decimal	Decimal
xsd:date	Date
xsd:time	Time
xsd:dateTime	DateTime
xsd:integer	Integer
gml:VectorType	Vector
gml:CodeType	GenericName (default), LocalName oder ScopeName
gml:LengthType	Length
gml:AngleType	Angle
gml:SpeedType	Velocity
gml:ScaleType	Scale
gml:AreaType	Area
gml:VolumeType	Volume
gml:MeasureType	Measure
xsd:nonPositiveInteger	Integer
xsd:negativeInteger	Integer
xsd:nonNegativeInteger	Integer
xsd:positiveInteger	Integer

#### F.2.3.6 GML Data Types

Every GML data type shall be mapped to a UML class. The name of the class shall be the same as the name of the complex type without the "Type"-suffix.

If the GML data type is derived from another GML data type (base type), then the class inherits from the corresponding superclass.

If the properties of the GML data type are embedded in a xsd:sequence element, the stereotype of the class shall be <<DataType>>, if they are embedded in a xsd:choice element, the stereotype of the class shall be set to <<Union>>.

The GML properties of the GML object type shall be mapped to attributes and association roles as described in clause F.2.3.9.

#### F.2.3.7 Enumerations

A simple type defined an the GML application schema as a restriction of xsd:string with enumeration values shall be mapped to a class with the <<Enumeration>> stereotype in the UML application schema.

The name of the class shall be the name of the simple type.

Every xsd:enumeration facet without an xsd:applInfo annotation with a source attribute "urn:x-gml:codelistValue" shall be mapped to an UML attribute with the value as the attribute name.

Every xsd:enumeration facet with an xsd:appInfo annotation with a source attribute “urn:x-gml:codelistValue” shall be mapped to an initial value of the UML attribute with the same name as the value of the appinfo element. If no such UML attribute exists in the class, the facet shall be ignored.

#### F.2.3.8 Code lists

A simple type defined an the GML application schema as a union of a xsd:pattern restriction with the value “other:w{2,}” and an enumeration shall be mapped to a class with the stereotype <>CodeList>> in the UML application schema.

The name of the class shall be the name of the simple type.

Every xsd:enumeration facet of the enumeration without an xsd:appInfo annotation with a source attribute “urn:x-gml:codelistValue” shall be mapped to an UML attribute with the value as the attribute name.

Every xsd:enumeration facet of the enumeration with an xsd:appInfo annotation with a source attribute “urn:x-gml:codelistValue” shall be mapped to an initial value of the UML attribute with the same name as the value of the appinfo element. If no such UML attribute exists in the class, the facet shall be ignored.

#### F.2.3.9 GML properties

If the type of a property element

- is a simple type or the property type of GML data type, the property shall be mapped to an UML attribute with the corresponding type as the data type
- is a by-value property type of a GML object type, the property shall be mapped to an UML association role of a UML composition to the class representing the target GML object type
- is a by-reference-or-value property type of a GML object type, the property shall be mapped to an UML association role of a UML association to the class representing the target GML object type
- is a by-reference property type of a GML object type, the property shall be mapped to an UML association role of a UML association to the class representing the target GML object type where the target GML object type shall be determined from the embedded xsd:appInfo annotation with a source attribute value “urn:x-gml:targetElement” specifying the qualified element name of the target type).

The name of the UML attribute or association role shall be the name of the GML property element.

The multiplicity of the UML attribute or association role shall be derived from the minOccurs and maxOccurs value of the GML property.

If the property element has an xsd:appInfo annotation with a source attribute value “urn:x-gml:reverseProperty” embedded, then the association role shall be defined as part of the association between the two classes where the other association role has a name equal to the value of the xsd:appInfo element.

#### F.2.3.10 Documentation

XML Schema xsd:annotation/xsd:documentation elements in GML application schemas are mapped to “documentation” tagged values in the UML application schema.

## Annex G (**Informative**)

### Guidelines for Subsetting GML Schemas

#### **G.1 General**

An automated approach is recommended for subsetting GML schemas. This annex contains an informative XSLT reference implementation of a GML schema subset tool. The tool consists of three XSLT stylesheets, the three stylesheets are show in clauses G.2, G.3 and G.4 below.

To create a GML subset schema using this tool:

1. Transform `gml.xsd` using `depends.xslt` and an XSLT processor to produce `gml.dep`, e.g. using Xalan by

```
$ java org.apache.xalan.xslt.Process -IN ..base/gml.xsd -XSL depends.xslt -OUT gml.dep
```

2. If the XSLT processor you are using cannot pass parameters to a stylesheet being processed, edit `gmlSubset.xslt`, and change the “wanted” parameter to contain a comma separated list (with a trailing comma) of the namespace-qualified global types and elements you want in your GML subset schema. For example, change

```
<xsl:param name="wanted">,</xsl:param>
```

to

```
<xsl:param name="wanted">gml:featureProperty,gml:lineStringProperty,gml:polygonProperty,</xsl:param>
```

3. Transform `gml.dep` using `gmlSubset.xslt`, a parameter named “wanted” set to a comma separated list (with a trailing comma) of the namespace qualified global types and elements you want in your GML subset schema, and an XSLT processor to produce `gmlSubset.xsd`, which will contain the global types and elements specified in the “wanted” parameter and all of the global types and elements on which they directly or indirectly use.

4. If your “wanted” list did not include or depend on any elements or types from the namespaces named “smil20” or `smil20lang`, you may wish to edit the generated schema to remove the following namespace definitions from the schema element:

```
xmlns:smil20="http://www.w3.org/2001/SMIL20/"
```

```
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language"
```

```
xmlns:x="http://www.w3.org/XML/1998/namespace"
```

5. The generated `gmlSubset.xsd` will include imports for the namespaces named “xlink”, “smil20” and “smil20lang” if your “wanted” list included or depended on any elements or types from the corresponding namespaces. Otherwise, it is a stand-alone GML subset schema that conforms to the requirements for GML profiles.

## G.2 depends.xslt

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xlink="http://www.w3.org/1999/xlink">
  <!-- =====
This stylesheet is designed to be used on gml.xsd to produce gml.dep
for use by the gml schema subset utility gmlSubset.xslt to produce a specialized
gmlSubset.xsd that contains only the specified types and elements, and the types
and elements on which they depend.
=====-->
<xsl:output method="xml" encoding="UTF-8" indent="yes"/>
<xsl:include href="utility.xslt"/>
<!-- NEWLINE = &#xA; -->
<xsl:param
name="schemas">gml.xsd,observation.xsd,dynamicFeature.xsd,coverage.xsd,topology.xsd,defaultStyle.xsd,coordinateReferenc
eSystems.xsd,feature.xsd,valueObjects.xsd,grids.xsd,geometryComplexes.xsd,datums.xsd,coordinateSystems.xsd,coordinateOp
erations.xsd,geometryAggregates.xsd,referenceSystems.xsd,dataQuality.xsd,geometryPrimitives.xsd,geometryBasic2d.xsd,direct
ion.xsd,geometryBasic0d1d.xsd,measures.xsd,temporal.xsd,units.xsd,dictionary.xsd,gmlBase.xsd,basicTypes.xsd,</xsl:param>
<xsl:param name="allSchemas">
  <xsl:call-template name="getUniqueSchemaList">
    <xsl:with-param name="list" select="$schemas"/>
    <xsl:with-param name="usePre"></xsl:with-param>
  </xsl:call-template>
</xsl:param>
<xsl:template match="/">
  <xsl:param name="docName">gml.xsd</xsl:param>
  <xsl:param name="top" select="true()"/>
  <xsl:param name="tns" select="//xsd:schema/@targetNamespace"/>
  <xsl:param name="vers" select="//xsd:schema/@version"/>
  <xsl:variable name="ltns">
    <xsl:for-each select="//xsd:schema/namespace::*">
      <xsl:if test="local-name() != 'targetNamespace' and string() = $tns">
        <xsl:value-of select="local-name()"/>
      </xsl:if>
    </xsl:for-each>
  </xsl:variable>
  <xsl:variable name="tnsp">
    <xsl:choose>
      <xsl:when test="$ltns = "">
        <xsl:call-template name="getTargetNameSpacePrefix">
          <xsl:with-param name="list" select="$tns"/>
        </xsl:call-template>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$ltns"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:text>&#xA;</xsl:text>
  <xsl:choose>
    <xsl:when test="$top">
      <xsl:text disable-output-escaping="yes">&lt;depends version="</xsl:text><xsl:value-of select="$vers"/><xsl:text
disable-output-escaping="yes">"&gt;</xsl:text>
    </xsl:when>
    <xsl:otherwise>
      <xsl:for-each select="/xsd:schema">
```

```

<xsl:for-each select="xsd:complexType | xsd:group | xsd:simpleType | xsd:element | xsd:attribute | xsd:attributeGroup">
    <xsl:variable name="type" select="local-name()"/>
    <xsl:choose>
        <xsl:when test="$type = 'complexType' ">
            <xsl:call-template name="complexType">
                <xsl:with-param name="docName" select="$docName"/>
                <xsl:with-param name="targetNamespace" select="$tnsp"/>
            </xsl:call-template>
        </xsl:when>
        <xsl:when test="$type = 'group' ">
            <xsl:call-template name="complexType">
                <xsl:with-param name="docName" select="$docName"/>
                <xsl:with-param name="targetNamespace" select="$tnsp"/>
            </xsl:call-template>
        </xsl:when>
        <xsl:when test="$type = 'simpleType' ">
            <xsl:call-template name="simpleType">
                <xsl:with-param name="docName" select="$docName"/>
                <xsl:with-param name="targetNamespace" select="$tnsp"/>
            </xsl:call-template>
        </xsl:when>
        <xsl:when test="$type = 'element' ">
            <xsl:call-template name="globalElement">
                <xsl:with-param name="docName" select="$docName"/>
                <xsl:with-param name="targetNamespace" select="$tnsp"/>
            </xsl:call-template>
        </xsl:when>
        <xsl:when test="$type = 'attribute' ">
            <xsl:call-template name="globalAtt">
                <xsl:with-param name="docName" select="$docName"/>
                <xsl:with-param name="targetNamespace" select="$tnsp"/>
            </xsl:call-template>
        </xsl:when>
        <xsl:when test="$type = 'attributeGroup' ">
            <xsl:call-template name="globalAtt">
                <xsl:with-param name="docName" select="$docName"/>
                <xsl:with-param name="targetNamespace" select="$tnsp"/>
            </xsl:call-template>
        </xsl:when>
        <xsl:otherwise/>
    </xsl:choose>
</xsl:for-each>
</xsl:otherwise>
</xsl:choose>
<xsl:if test="$top">
    <xsl:call-template name="dependSchemas">
        <xsl:with-param name="list" select="$allSchemas"/>
    </xsl:call-template>
    <xsl:text disable-output-escaping="yes">&#xA;&lt;/depends&gt;&#xA;</xsl:text>
</xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template name="complexType">
    <xsl:param name="docName"/>
    <xsl:param name="targetNamespace"/>
    <xsl:variable name="name" select="@name"/>
    <xsl:if test="$name">
        <xsl:element name="def">

```

```

<xsl:attribute name="name"><xsl:value-of select="$targetNamespace"/><:xsl:value-of
select="$name"/></xsl:attribute>
<xsl:attribute name="doc"><xsl:value-of select="$docName"/></xsl:attribute>
<xsl:variable name="uses">
    <xsl:apply-templates select="./*[xsd:complexType|xsd:simpleContent]"/>
    <xsl:call-template name="ElAndAtt"/>
</xsl:variable>
<!-- USES <xsl:value-of select="$uses"/> -->
<xsl:call-template name="writeUses">
    <xsl:with-param name="list" select="$uses"/>
</xsl:call-template>
</xsl:element>
</xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template match="xsd:complexContent">
    <xsl:for-each select="descendant::xsd:extension">
        <xsl:value-of select="@base"/>
        <xsl:text>?extension|</xsl:text>
    </xsl:for-each>
    <xsl:for-each select="descendant::xsd:restriction">
        <xsl:value-of select="@base"/>
        <xsl:text>?restriction|</xsl:text>
    </xsl:for-each>
</xsl:template>
<!-- ===== -->
<xsl:template match="xsd:simpleContent">
    <xsl:for-each select="descendant::xsd:extension">
        <xsl:value-of select="@base"/>
        <xsl:text>?extension|</xsl:text>
    </xsl:for-each>
    <xsl:for-each select="descendant::xsd:restriction">
        <xsl:value-of select="@base"/>
        <xsl:text>?restriction|</xsl:text>
    </xsl:for-each>
</xsl:template>
<!-- ===== -->
<xsl:template name="ElAndAtt">
    <xsl:for-each select="descendant::xsd:element | descendant::xsd:group | descendant::xsd:attribute | descendant::xsd:attributeGroup">
        <xsl:variable name="name" select="@type | @ref"/>
        <xsl:if test="$name and contains($name,':')">
            <xsl:value-of select="$name"/>
            <xsl:text>|</xsl:text>
        </xsl:if>
    </xsl:for-each>
</xsl:template>
<!-- ===== -->
<xsl:template name="simpleType">
    <xsl:param name="docName"/>
    <xsl:param name="targetNamespace"/>
    <xsl:variable name="name" select="@name"/>
    <xsl:if test="$name">
        <xsl:element name="def">
            <xsl:attribute name="name"><xsl:value-of select="$targetNamespace"/><:xsl:value-of
select="$name"/></xsl:attribute>
            <xsl:attribute name="doc"><xsl:value-of select="$docName"/></xsl:attribute>
            <!-- SIMPLE <xsl:copy-of select="./*"/>-->
            <xsl:variable name="uses">

```

```

<xsl:for-each select="xsd:union">
  <!-- UNION <xsl:value-of select="@memberTypes"/> -->
    <xsl:variable name="members" select="@memberTypes"/>
    <xsl:if test="$members">
      <xsl:value-of select="translate($members, ',', '|')"/>
      <xsl:text>|</xsl:text>
    </xsl:if>
  </xsl:for-each>
  <xsl:for-each select="xsd:list">
    <xsl:variable name="items" select="@itemType"/>
    <xsl:if test="$items">
      <xsl:value-of select="$items"/>
      <xsl:text>|</xsl:text>
    </xsl:if>
  </xsl:for-each>
  </xsl:variable>
  <!-- USES <xsl:value-of select="$uses"/> -->
  <xsl:call-template name="writeUses">
    <xsl:with-param name="list" select="$uses"/>
  </xsl:call-template>
  </xsl:element>
</xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template name="globalElement">
  <xsl:param name="docName"/>
  <xsl:param name="targetNamespace"/>
  <xsl:variable name="name" select="@name"/>
  <xsl:if test="$name">
    <xsl:element name="def">
      <xsl:attribute name="name"><xsl:value-of select="$targetNamespace"/></xsl:value-of
select="$name"/></xsl:attribute>
      <xsl:attribute name="doc"><xsl:value-of select="$docName"/></xsl:attribute>
      <xsl:variable name="uses">
        <xsl:variable name="type" select="@type"/>
        <xsl:if test="$type and contains($type,':')">
          <xsl:value-of select="$type"/>
          <xsl:text>|</xsl:text>
        </xsl:if>
        <xsl:variable name="sub" select="@substitutionGroup"/>
        <xsl:if test="$sub">
          <xsl:value-of select="$sub"/>
          <xsl:text>|</xsl:text>
        </xsl:if>
      </xsl:variable>
      <!-- USES <xsl:value-of select="$uses"/> -->
      <xsl:call-template name="writeUses">
        <xsl:with-param name="list" select="$uses"/>
      </xsl:call-template>
    </xsl:element>
  </xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template name="globalAtt">
  <xsl:param name="docName"/>
  <xsl:param name="targetNamespace"/>
  <xsl:variable name="name" select="@name"/>
  <xsl:if test="$name">
    <xsl:element name="def">

```

```

<xsl:attribute name="name"><xsl:value-of select="$targetNamespace"/></xsl:value-of>
<select="$name"/></xsl:attribute>
<xsl:attribute name="doc"><xsl:value-of select="$docName"/></xsl:attribute>
<xsl:variable name="uses">
    <xsl:variable name="type" select="@type"/>
    <xsl:if test="$type and contains($type,':')">
        <xsl:value-of select="$type"/>
        <xsl:text>|</xsl:text>
    </xsl:if>
    <xsl:call-template name="ElAndAtt"/>
</xsl:variable>
<!-- USES <xsl:value-of select="$uses"/> -->
<xsl:call-template name="writeUses">
    <xsl:with-param name="list" select="$uses"/>
</xsl:call-template>
</xsl:element>
</xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template name="writeUses">
    <xsl:param name="list"/>
    <xsl:if test="$list != "">
        <xsl:variable name="first" select="substring-before($list, '|')"/>
        <xsl:variable name="eor" select="substring-after($first, '?')"/>
        <xsl:variable name="use">
            <xsl:choose>
                <xsl:when test="contains($first, '?')">
                    <xsl:value-of select="substring-before($first, '?')"/>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:value-of select="$first"/>
                </xsl:otherwise>
            </xsl:choose>
        </xsl:variable>
        <xsl:variable name="testp">
            <xsl:value-of select="$use"/>
            <xsl:text>|</xsl:text>
        </xsl:variable>
        <xsl:variable name="testq">
            <xsl:value-of select="$use"/>
            <xsl:text>?</xsl:text>
        </xsl:variable>
        <xsl:variable name="rest" select="substring-after($list, '|')"/>
        <xsl:choose>
            <xsl:when test="contains($rest, $testp)">
            <xsl:when test="contains($rest, $testq)">
            <xsl:when test="$use = """/>
            <xsl:otherwise>
                <xsl:element name="uses">
                    <xsl:attribute name="name"><xsl:value-of select="$use"/></xsl:attribute>
                    <xsl:if test="$eor != "">
                        <xsl:attribute name="derivation"><xsl:value-of select="$eor"/> </xsl:attribute>
                    </xsl:if>
                </xsl:element>
            </xsl:otherwise>
        </xsl:choose>
        <xsl:call-template name="writeUses">
            <xsl:with-param name="list" select="$rest"/>
        </xsl:call-template>
    </xsl:if>
</xsl:template>

```

```

</xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template name="dependSchemas">
  <xsl:param name="list"/>
  <xsl:if test="$list != "">
    <xsl:variable name="first" select="substring-before($list, ',')"/>
    <xsl:variable name="rest" select="substring-after($list, ',')"/>
    <xsl:apply-templates select="document($first, /)">
      <xsl:with-param name="docName" select="$first"/>
      <xsl:with-param name="top" select="false()"/>
    </xsl:apply-templates>
    <xsl:choose>
      <xsl:when test="contains($rest,',')">
        <xsl:call-template name="dependSchemas">
          <xsl:with-param name="list" select="$rest"/>
        </xsl:call-template>
      </xsl:when>
      <xsl:otherwise/>
    </xsl:choose>
  </xsl:if>
</xsl:template>
<!-- ===== -->
<!-- ===== -->
</xsl:stylesheet>

```

### G.3 gmlSubset.xslt

```

<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xlink="http://www.w3.org/1999/xlink">
  <xsl:output method="xml" encoding="UTF-8" indent="yes"/>
<!-- ===== -->
This stylesheet is designed to be used on gml.dep (produced from
gml.xsd by depends.xslt) to produce a specialized gmlSubset.xsd that
contains only the types and elements specified in the "wanted" parameter,
and the types and elements on which they depend. Note that the type and
element items in the "wanted" parameter must include namespace prefixes,
and that they must be separated by commas, including a trailing comma after
the last item.
<===== -->
<xsl:include href="utility.xslt"/>
<xsl:param name="baseUri" select="document('../base/gml.xsd')"/>
<!-- sample1 <xsl:param name="wanted">gml:featureProperty,gml:lineStringProperty,gml:polygonProperty,</xsl:param> -->
<!-- sample2 <xsl:param name="wanted">gml:GeographicCRS,gml:_Coverage,gml:track,</xsl:param> -->
<!-- sample3 <xsl:param
name="wanted">gml:_FeatureCollection,gml:itemStyleDescriptorType,gml:FeatureConstraintType,</xsl:param> -->
<xsl:param
name="wanted">gml:metaDataProperty,gml:_association,gml:members,gml:Array,gml:curveProperty,gml:LineString,gml:LinearRing,gml:exterior,gml:interior,gml:surfaceMember,gml:surfaceProperty,gml:multiSurfaceProperty,gml:directedNode,gml:directedEdge,gml:directedFace,gml:isolatedProperty,gml:featureProperty,gml:featureMembers,gml:_FeatureCollection,gml:featureMember,gml:BaseStyleDescriptorType,</xsl:param>
<xsl:template match="/">
  <xsl:variable name="wantedList">
    <xsl:call-template name="getWantedList">
      <xsl:with-param name="list" select="$wanted"/>
      <xsl:with-param name="from" value="BEGIN"><xsl:with-param>
      <xsl:with-param name="depth" value="0"><xsl:with-param>
    </xsl:call-template>

```

```

</xsl:variable>
<xsl:variable name="vers" select="//depends/@version"/>
<schema targetNamespace="http://www.opengis.net/gml" xmlns="http://www.w3.org/2001/XMLSchema"
xmins:sch="http://www.ascc.net/xml/schematron" xmlns:gml="http://www.opengis.net/gml"
xmins:xlink="http://www.w3.org/1999/xlink" xmlns:smil20="http://www.w3.org/2001/SMIL20"
xmlns:smil20lang="http://www.w3.org/2001/SMIL20/Language" xmlns:x="http://www.w3.org/XML/1998/namespace"
elementFormDefault="qualified" version="${vers}">
  <annotation>
    <documentation>GML Subset schema for <xsl:value-of select="$wanted"/> written by gmlSubset.xslt.
  </documentation>
  </annotation>

  <xsl:if test="contains($wantedList,'xlink:')">
    <import namespace="http://www.w3.org/1999/xlink" schemaLocation="../../xlink/xlinks.xsd"/>
  </xsl:if>
  <xsl:if test="contains($wantedList,'smil20:')">
    <import namespace="http://www.w3.org/2001/SMIL20/" schemaLocation="../../smil/smil20.xsd"/>
  </xsl:if>
  <xsl:if test="contains($wantedList,'smil20lang:')">
    <import namespace="http://www.w3.org/2001/SMIL20/Language" schemaLocation="../../smil/smil20-
language.xsd"/>
  </xsl:if>
  <xsl:call-template name="writeWantedList">
    <xsl:with-param name="list" select="$wantedList"/>
  </xsl:call-template>
</schema>
</xsl:template>
<!-- ===== -->
<xsl:template name="getDocName">
  <xsl:param name="wanted"/>
  <xsl:for-each select="//depends/def[@name=$wanted]">
    <xsl:value-of select="@doc"/>
  </xsl:for-each>
</xsl:template>
<!-- ===== -->
<xsl:template name="getUses">
  <xsl:param name="wanted"/>
  <xsl:for-each select="//depends/def[@name=$wanted]">
    <xsl:for-each select="uses">
      <xsl:value-of select="@name"/>
      <xsl:text>,</xsl:text>
    </xsl:for-each>
  </xsl:for-each>
</xsl:template>
<!-- ===== -->
<xsl:template name="writeWanted">
  <xsl:param name="wanted"/>
  <xsl:choose>
    <xsl:when test="contains($wanted,'xlink:') or contains($wanted,'wfs:') or contains($wanted,'smil20')">
      <!-- XLINK <xsl:value-of select="$wanted"/> -->
    </xsl:when>
    <xsl:otherwise>
      <!-- OTHER <xsl:value-of select="$wanted"/> -->
      <xsl:variable name="docName">
        <xsl:call-template name="getDocName">
          <xsl:with-param name="wanted" select="$wanted"/>
        </xsl:call-template>
      </xsl:variable>
      <xsl:variable name="localName">

```

```

<xsl:call-template name="removePrefix">
  <xsl:with-param name="name" select="$wanted"/>
  <xsl:with-param name="pre"></xsl:with-param>
</xsl:call-template>
</xsl:variable>
<xsl:call-template name="Separator"/>
<xsl:for-each select="document($docName,$baseUri)">
  <xsl:for-each select="//xsd:schema/*[@name = $localName]">
    <xsl:copy-of select=". "/>
  </xsl:for-each>
</xsl:for-each>
</xsl:otherwise>
</xsl:choose>
</xsl:template>
<!-- ===== -->
<xsl:template name="writeWantedList">
  <xsl:param name="list"/>
  <xsl:if test="$list != "">
    <xsl:variable name="first" select="substring-before($list, ',')"/>
    <xsl:variable name="rest" select="substring-after($list, ',')"/>
    <xsl:call-template name="writeWanted">
      <xsl:with-param name="wanted" select="$first"/>
    </xsl:call-template>
    <xsl:if test="contains($rest,',')">
      <xsl:call-template name="writeWantedList">
        <xsl:with-param name="list" select="$rest"/>
      </xsl:call-template>
    </xsl:if>
  </xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template name="getWantedList">
  <xsl:param name="list"/>
  <xsl:param name="seen"/>
  <xsl:param name="from"/>
  <xsl:param name="depth"/>
  <xsl:if test="$list != "">
    <xsl:variable name="first" select="substring-before($list, ',')"/>
    <xsl:variable name="firstSep" select="concat($first,',')"/>
    <xsl:variable name="rest" select="substring-after($list, ',')"/>
    <xsl:choose>
      <xsl:when test="contains($seen,$firstSep)">
        <xsl:call-template name="getWantedList">
          <xsl:with-param name="list" select="$rest"/>
          <xsl:with-param name="seen" select="$seen"/>
          <xsl:with-param name="from">REST</xsl:with-param>
          <xsl:with-param name="depth" select="$depth + 1"/>
        </xsl:call-template>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$firstSep"/>
        <xsl:variable name="usesList">
          <xsl:call-template name="getUses">
            <xsl:with-param name="wanted" select="$first"/>
          </xsl:call-template>
        </xsl:variable>
        <xsl:variable name="ToDo" select="concat($usesList,$rest)"/>
        <xsl:variable name="nowSeen" select="concat($seen,$firstSep)"/>
        <xsl:call-template name="getWantedList">
          <xsl:with-param name="list" select="$ToDo"/>

```

```

        <xsl:with-param name="seen" select="$nowSeen"/>
        <xsl:with-param name="from">USES</xsl:with-param>
        <xsl:with-param name="depth" select="$depth + 1"/>
    </xsl:call-template>
    </xsl:otherwise>
</xsl:choose>
</xsl:if>
</xsl:template>
<!-- ===== -->
</xsl:stylesheet>

```

## G.4 utility.xslt

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsl:output method="xml" encoding="UTF-8" indent="yes"/>
    <!-- ===== -->
    <xsl:template name="getTargetNameSpacePrefix">
        <xsl:param name="list"/>
        <xsl:if test="$list != "">
            <xsl:variable name="first" select="substring-before($list, '/')"/>
            <xsl:variable name="rest" select="substring-after($list, '/')"/>
            <xsl:choose>
                <xsl:when test="contains($rest,'/')>
                    <xsl:call-template name="getTargetNameSpacePrefix">
                        <xsl:with-param name="list" select="$rest"/>
                    </xsl:call-template>
                </xsl:when>
                <xsl:when test="$rest = "">
                    <xsl:value-of select="$first"/>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:value-of select="$rest"/>
                </xsl:otherwise>
            </xsl:choose>
        </xsl:if>
    </xsl:template>
    <!-- ===== -->
    <xsl:template name="getPathPrefix">
        <xsl:param name="file"/>
        <xsl:if test="contains($file,'/')>
            <xsl:variable name="pre" select="substring-before($file,'/')"/>
            <xsl:variable name="suf" select="substring-after($file,'/')"/>
            <xsl:choose>
                <xsl:when test="contains($suf,'/')>
                    <xsl:value-of select="$pre"/><xsl:text></xsl:text>
                    <xsl:call-template name="getPathPrefix">
                        <xsl:with-param name="file" select="$suf"/>
                    </xsl:call-template>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:variable name="path">
                        <xsl:call-template name="removeSuffix">
                            <xsl:with-param name="name" select="$file"/>
                            <xsl:with-param name="suf" select="$suf"/>
                        </xsl:call-template>
                    </xsl:variable>
                    <xsl:value-of select="$path"/>
                </xsl:otherwise>
            </xsl:choose>
        </xsl:if>
    </xsl:template>

```

```

        </xsl:otherwise>
    </xsl:choose>
</xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template name="removePrefix">
    <xsl:param name="name"/>
    <xsl:param name="pre"/>
    <xsl:variable name="npName">
        <xsl:choose>
            <xsl:when test="contains($name,$pre)">
                <xsl:value-of select="substring-after($name,$pre)"/>
            </xsl:when>
            <xsl:otherwise>
                <xsl:value-of select="$name"/>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:variable>
    <xsl:value-of select="$npName"/>
</xsl:template>
<!-- ===== -->
<xsl:template name="removeSuffix">
    <xsl:param name="name"/>
    <xsl:param name="suf"/>
    <xsl:variable name="nsName">
        <xsl:choose>
            <xsl:when test="contains($name,$suf)">
                <xsl:value-of select="substring-before($name,$suf)"/>
            </xsl:when>
            <xsl:otherwise>
                <xsl:value-of select="$name"/>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:variable>
    <xsl:value-of select="$nsName"/>
</xsl:template>
<!-- ===== -->
<xsl:template name="lowerLeading">
    <xsl:param name="name"/>
    <xsl:variable name="ch1" select="substring($name, 1, 1)"/>
    <xsl:variable name="lc1" select="translate($ch1,'ABCDEFGHIJKLMNOPQRSTUVWXYZ','abcdefghijklmnopqrstuvwxyz')"/>
    <xsl:value-of select="concat($lc1, substring($name, 2))"/>
</xsl:template>
<!-- ===== -->
<xsl:template name="uniqueList">
    <xsl:param name="list"/>
    <xsl:param name="sep"/>
    <xsl:param name="seen"/>
    <xsl:param name="pre"../base/<xsl:param>
    <xsl:if test="$list != "">
        <xsl:variable name="first" select="substring-before($list, $sep)"/>
        <xsl:variable name="firstSep" select="concat($first,$sep)"/>
        <xsl:variable name="rest" select="substring-after($list, $sep)"/>
        <xsl:choose>
            <xsl:when test="contains($seen,$firstSep)">
                <xsl:call-template name="uniqueList">
                    <xsl:with-param name="list" select="$rest"/>
                    <xsl:with-param name="sep" select="$sep"/>
                    <xsl:with-param name="seen" select="$seen"/>
            </xsl:when>
        </xsl:choose>
    </xsl:if>
</xsl:template>

```

```

        </xsl:call-template>
    </xsl:when>
    <xsl:otherwise>
        <xsl:value-of select="$firstSep"/>
        <xsl:variable name="nowSeen" select="concat($seen, $firstSep)" />
        <xsl:call-template name="uniqueList">
            <xsl:with-param name="list" select="$rest"/>
            <xsl:with-param name="sep" select="$sep"/>
            <xsl:with-param name="seen" select="$nowSeen"/>
            </xsl:call-template>
        </xsl:otherwise>
    </xsl:choose>
</xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template name="getIncludedDocs">
    <xsl:param name="docName"/>
    <xsl:param name="usePre"/>
    <xsl:param name="seenList"/>
    <xsl:param name="sep">,</xsl:param>
    <xsl:value-of select="$docName"/>
    <xsl:text>,</xsl:text>
    <xsl:variable name="pathPre">
        <xsl:call-template name="getPathPrefix">
            <xsl:with-param name="file" select="$docName"/>
        </xsl:call-template>
    </xsl:variable>
    <xsl:variable name="callPathPre">
        <xsl:choose>
            <xsl:when test="$pathPre = " or $pathPre = './ ">
                <xsl:value-of select="$usePre"/>
            </xsl:when>
            <xsl:otherwise>
                <xsl:value-of select="$pathPre"/>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:variable>
    <xsl:for-each select="document($docName, / )">
        <xsl:for-each select="//xsd:include | //xsd:import">
            <xsl:variable name="iDoc" select="@schemaLocation"/>
            <xsl:variable name="iPathPre">
                <xsl:call-template name="getPathPrefix">
                    <xsl:with-param name="file" select="$iDoc"/>
                </xsl:call-template>
            </xsl:variable>
            <xsl:variable name="iDocSuf">
                <xsl:call-template name="removePrefix">
                    <xsl:with-param name="name" select="$iDoc"/>
                    <xsl:with-param name="pre" select="$iPathPre"/>
                </xsl:call-template>
            </xsl:variable>
            <xsl:variable name="usePathPre">
                <xsl:choose>
                    <xsl:when test="$iPathPre = " or $iPathPre = './ ">
                        <xsl:value-of select="$callPathPre"/>
                    </xsl:when>
                    <xsl:otherwise>
                        <xsl:value-of select="$iPathPre"/>
                    </xsl:otherwise>
                </xsl:choose>
            </xsl:variable>
        </xsl:for-each>
    </xsl:for-each>

```

```

        </xsl:choose>
    </xsl:variable>
    <xsl:variable name="uDoc">
        <xsl:value-of select="concat($usePathPre,$iDocSuf)" />
    </xsl:variable>
    <xsl:variable name="uDocSep">
        <xsl:value-of select="concat($uDoc,$sep)" />
    </xsl:variable>
    <xsl:if test="not(contains($seenList,$uDocSep))">
        <xsl:variable name="seenListPlus" select="concat($seenList,$uDocSep)" />
        <xsl:call-template name="getIncludedDocs">
            <xsl:with-param name="docName" select="$uDoc" />
            <xsl:with-param name="usePre" select="$usePathPre" />
            <xsl:with-param name="seenList" select="$seenListPlus" />
        </xsl:call-template>
    </xsl:if>
    </xsl:for-each>
</xsl:for-each>
</xsl:template>
<!-- ===== -->
<xsl:template name="getDocumentList">
    <xsl:param name="list" />
    <xsl:param name="seenList" />
    <xsl:param name="usePre" />
    <xsl:if test="$list != "">
        <xsl:variable name="first" select="substring-before($list, ',')"/>
        <xsl:variable name="rest" select="substring-after($list, ',')"/>
        <xsl:variable name="included">
            <xsl:call-template name="getIncludedDocs">
                <xsl:with-param name="docName" select="$first" />
                <xsl:with-param name="usePre" select="$usePre" />
                <xsl:with-param name="seenList" select="$seenList" />
            </xsl:call-template>
        </xsl:variable>
        <xsl:value-of select="$included" />
        <xsl:variable name="seenListIncluded" select="concat($seenList,$included)" />
        <xsl:if test="contains($rest, ',')">
            <xsl:call-template name="getDocumentList">
                <xsl:with-param name="list" select="$rest" />
                <xsl:with-param name="seenList" select="$seenListIncluded" />
                <xsl:with-param name="usePre" select="$usePre" />
            </xsl:call-template>
        </xsl:if>
    </xsl:if>
</xsl:template>
<!-- ===== -->
<xsl:template name="getUniqueSchemaList">
    <xsl:param name="list" />
    <xsl:param name="usePre" />
    <xsl:variable name="allSchemas">
        <xsl:call-template name="getDocumentList">
            <xsl:with-param name="list" select="$list" />
            <xsl:with-param name="usePre" select="$usePre" />
        </xsl:call-template>
    </xsl:variable>
    <xsl:variable name="uniqueSchemas">
        <xsl:call-template name="uniqueList">
            <xsl:with-param name="list" select="$allSchemas" />
            <xsl:with-param name="sep" />, </xsl:with-param>
        </xsl:call-template>
    </xsl:variable>

```

```
</xsl:variable>
<xsl:value-of select="$uniqueSchemas"/>
</xsl:template>
<!-- ===== -->
<xsl:template name="Separator">
<xsl:param name="comment" select="====="/>
<xsl:text>&#xA;</xsl:text>
<xsl:text disable-output-escaping="yes">
  &lt;!-- =====<xsl:text><xsl:value-of select="$comment"/><xsl:text disable-output-
escaping="yes">===== --&gt;
</xsl:text>
<!-- ===== -->
</xsl:template>
</xsl:stylesheet>
```

## Bibliography

ISO 31 (all parts), *Quantities and units*.

ISO 1000, *SI units and recommendations for the use of their multiples and of certain other units*.

Cover Pages Technology Report at <http://xml.coverpages.org/geographyML.html>

Langran, G. *Time in Geographic Information Systems*. London: Taylor & Francis Ltd. 1992.

Kaufman, M. & Wagner D. *Drawing Graphs*, Springer LNCS 2025, 1998.

Battista, G., Eades P., Tamassia R. & Tollis I. *Graph Drawing*, Prentice Hall 1999.

## Index

—  
\_association 24, 25, 26, 607  
\_ContinuousCoverage 248, 249  
\_CoordinateOperation 150, 152, 155, 310, 311  
\_CoordinateReferenceSystem 111, 112, 114, 115, 116, 118, 119, 120, 326, 327, 328, 329, 332, 333  
\_CoordinateSystem 127, 128, 129, 130, 131, 132, 133, 134, 338, 339, 340, 341, 342, 343  
\_Coverage 14, 24, 247, 248, 263, 344, 606  
\_CRS 24, 107, 108, 111, 113, 325, 326, 453  
\_Curve 49, 50, 55, 56, 57, 58, 60, 63, 64, 65, 68, 69, 71, 72, 76, 88, 395, 404, 407, 409, 410, 412, 415, 416, 418, 420, 421, 422, 575, 579, 583  
\_CurveSegment 55, 56, 57, 58, 60, 63, 64, 65, 69, 71, 72, 409, 410, 412, 415, 416, 418, 420, 421, 422, 575, 583  
\_Datum 137, 139, 143, 144, 145, 146, 356, 357, 358, 359  
\_DiscreteCoverage 248, 348, 349, 350, 351  
\_Feature 13, 24, 32, 35, 39, 40, 41, 239, 240, 248, 249, 268, 284, 344, 378, 450, 574, 581, 582, 607  
\_FeatureCollection 40, 607  
\_GeneralConversion 153, 154, 157, 313  
\_GeneralDerivedCRS 112, 116, 117, 330, 331  
\_GeneralOperationParameter 164, 165, 166, 321, 323  
\_generalParameterValue 159, 161, 162, 316, 318  
\_GeneralTransformation 154, 155, 158, 315  
\_GeometricAggregate 90, 91, 92, 93, 94, 96, 380, 381, 383, 384, 385, 386  
\_GeometricPrimitive 47, 48, 49, 51, 85, 392, 393, 394, 399, 435  
\_Geometry 13, 34, 43, 44, 48, 89, 90, 232, 240, 249, 284, 345, 379, 380, 390, 391, 392, 398, 401, 406, 444, 450, 481, 579  
\_GML 13, 23, 24, 29, 32, 43, 183, 196, 204, 207, 268, 269, 271, 272, 273, 276, 284, 364, 365, 366, 367, 368, 371, 375, 377, 390, 440, 455, 461  
\_GriddedSurface 78, 80, 430, 431  
\_ImplicitGeometry 244, 444  
\_MetaData 13, 30, 31, 441, 443  
\_Object 23, 24, 25, 27, 29, 30, 172, 231, 240, 439, 440, 442, 470, 482  
\_Operation 152, 153, 154, 312, 313, 314  
\_ParametricCurveSurface 77, 79, 429  
\_positionalAccuracy 150, 153, 154, 169, 170, 309, 313, 315, 352, 353  
\_reference 26, 27  
\_ReferenceSystem 13, 106, 107, 452  
\_Ring 53, 76, 402, 428  
\_ScalarValue 229, 230, 231, 236, 250, 292, 345, 482, 483, 486  
\_ScalarValueList 230, 231, 250, 345, 482, 483  
\_SingleOperation 152, 156, 311, 312  
\_Solid 85, 86, 89, 405, 436, 579  
\_strictAssociation 26, 441  
\_Style 267, 268, 269, 364, 536, 541  
\_Surface 51, 52, 73, 74, 75, 77, 88, 400, 401, 404, 425, 426, 427, 428, 429, 579  
\_SurfacePatch 73, 74, 75, 77, 427, 428, 429  
\_TimeComplex 184, 185, 193, 466  
\_TimeGeometricPrimitive 185, 186, 458  
\_timeLength 190, 225, 458, 459  
\_TimeObject 183, 185, 232, 249, 284, 345, 456, 457, 481  
\_TimePrimitive 183, 184, 185, 192, 456, 457, 467  
\_TimeReferenceSystem 462, 464, 465  
\_TimeSlice 204, 206, 375, 376  
\_TimeTopologyPrimitive 192, 193, 468  
\_Topology 13, 24, 172, 180, 470, 476  
\_TopoPrimitive 172, 173, 174, 175, 181, 471, 472, 473, 477  
\_Value 24, 231, 232, 235, 251, 292, 346, 481, 482, 483, 484, 485

**0**

0D 10

**1**

19100 xxi, 1, 12, 22, 188, 249, 490, 491, 547, 562, 565, 579, 593, 595, 597  
 19103 2, 3, 5, 6, 22, 224, 445, 478, 491, 492, 547, 549, 562, 564, 565, 567, 571, 585, 592  
 19107 2, 3, 4, 5, 6, 7, 8, 41, 50, 52, 57, 74, 87, 171, 395, 401, 409, 427, 492, 493, 495, 496, 501, 504, 508, 511, 512, 513, 514, 516, 520, 521  
 19108 2, 182, 183, 184, 188, 189, 191, 196, 455, 460, 461, 463, 466, 521, 522, 523, 524, 525, 526, 527, 528, 529  
 19109 2, 7, 21, 31, 39, 283, 287, 490, 529, 530, 562, 563, 564, 565, 569, 579, 585, 592  
 19111 3, 4, 5, 6, 98, 99, 103, 111, 124, 136, 150, 169, 308, 325, 335, 351, 354, 451, 530, 531, 532  
 19112 491, 532  
 19115 2, 31, 103, 168, 275, 451, 527, 533, 568, 580, 592  
 19117 2, 491, 533, 534, 535, 536, 537, 538, 541, 542  
 19118 1, 2, 445, 478, 490, 491, 530, 562, 567, 569, 577, 592  
 19123 2, 5, 6, 7, 136, 242, 248, 249, 255, 257, 344, 347, 544, 545, 546, 547  
 19139 2, 31, 533, 568, 580, 592  
 1D 10, 14, 109, 110, 115, 120, 327, 333

**2**

2D 10, 12, 14, 52, 60, 62, 63, 64, 69, 70, 78, 79, 98, 109, 110, 117, 121, 123, 172, 173, 330, 401, 413, 414, 415, 416, 417, 430, 472

**3**

3D 10, 12, 14, 60, 62, 69, 70, 78, 81, 98, 99, 100, 108, 109, 115, 116, 123, 124, 171, 172, 176, 179, 327, 328, 413, 414, 417, 432, 475, 531

**A**

absoluteExternalPositionalAccuracy 169  
 AbsoluteExternalPositionalAccuracyType 169, 352  
 abstract 1, 8, 12, 22, 23, 24, 26, 27, 30, 32, 35, 39, 40, 42, 43, 44, 47, 48, 49, 51, 53, 55, 56, 73, 77, 79, 85, 90, 101, 102, 106, 107, 108, 110, 111, 112, 125, 127, 137, 140, 142, 146, 149, 150, 152, 153, 154, 155, 159, 162, 164, 165, 166, 169, 172, 183, 185, 190, 192, 193, 203, 204, 206, 208, 229, 231, 247, 248, 249, 268, 269, 284, 286, 291, 301, 308, 311, 312, 313, 314, 315, 316, 319, 321, 323, 325, 329, 335, 337, 344, 352, 354, 355, 358, 360, 361, 364, 371, 375, 377, 378, 379, 380, 389, 390, 391, 392, 394, 399, 400, 401, 408, 426, 429, 435, 438, 439, 440, 441, 442, 444, 451, 452, 453, 455, 456, 457, 459, 461, 462, 467, 470, 482, 492, 493, 495, 497, 498, 499, 503, 508, 509, 511, 512, 524, 531, 536, 537, 538, 541, 544, 564, 566, 573, 574, 578, 587, 589, 595  
 Abstract Specification xxi, 1, 2, 31, 39, 98, 103, 111, 124, 136, 150, 168, 171, 242, 307, 309, 325, 332, 335, 351, 354, 356, 451  
 AbstractContinuousCoverageType 248, 249, 292, 344  
 AbstractCoordinateOperationBaseType 150, 308  
 AbstractCoordinateOperationType 150, 152, 155, 308, 310, 311  
 AbstractCoordinateReferenceSystemType 111, 112, 114, 115, 116, 118, 119, 120, 325, 327, 328, 329, 332, 333, 334  
 AbstractCoordinateSystem BaseType 127, 337  
 AbstractCoordinateSystemType 127, 128, 129, 130, 131, 132, 133, 134, 337, 338, 339, 340, 341, 342, 343  
 AbstractCoverageType 247, 248, 249, 258, 259, 261, 262, 263, 264, 344  
 AbstractCRSType 107, 111, 113, 325, 326, 452  
 AbstractCurveSegmentType 55, 56, 57, 59, 62, 64, 65, 69, 70, 71, 408, 409, 410, 412, 415, 416, 419, 420, 421, 423, 575, 583, 588  
 AbstractCurveType 49, 50, 55, 68, 88, 394, 395, 404, 407, 408, 579, 588, 595  
 AbstractDatumBaseType 137, 355  
 AbstractDatumType 137, 139, 143, 144, 145, 146, 354, 356, 357, 358, 359  
 AbstractDiscreteCoverageType 248, 257, 292, 344, 348, 349, 350, 351  
 AbstractFeatureCollectionType 40, 41, 286, 378, 379, 573, 589, 591  
 AbstractFeatureType 25, 28, 32, 39, 40, 41, 203, 239, 268, 270, 286, 287, 292, 301, 344, 376, 377, 378, 379, 450, 570, 573, 574, 581, 582, 587, 589, 593, 595  
 AbstractGeneralConversionType 153, 157, 313, 314  
 AbstractGeneralDerivedCRSType 112, 117, 329, 330, 331  
 abstractGeneralOperationParameterRef 164  
 AbstractGeneralOperationParameterRefType 163, 164, 166, 167, 168, 320, 321, 322, 324  
 AbstractGeneralOperationParameterType 164, 165, 166, 321, 323  
 AbstractGeneralParameterValueType 159, 161, 162, 316, 317, 318  
 AbstractGeneralTransformationType 154, 158, 314, 316

AbstractGeometricAggregateType 87, 90, 91, 92, 94, 95, 96, 380, 381, 382, 383, 384, 385, 386, 387  
AbstractGeometricPrimitiveType 42, 47, 48, 49, 51, 85, 87, 392, 393, 394, 400, 435, 588  
AbstractGeometryType 33, 41, 42, 43, 44, 47, 53, 89, 90, 245, 288, 301, 380, 390, 392, 401, 406, 444, 579, 588, 595  
AbstractGMLType 7, 23, 24, 25, 27, 28, 29, 32, 39, 42, 172, 183, 196, 198, 204, 207, 208, 228, 232, 268, 269, 294, 301, 364, 365, 371, 375, 377, 391, 439, 440, 456, 470, 483, 570, 572, 574, 587, 588, 590, 593, 594  
AbstractGriddedSurfaceType 78, 79, 80, 429, 431, 432  
AbstractMetaDataType 30, 31, 440, 441  
AbstractOperationType 152, 153, 154, 312, 313, 315  
AbstractParametricCurveSurfaceType 77, 78, 429, 430  
AbstractPositionalAccuracyType 169, 170, 352, 353  
AbstractReferenceSystem BaseType 106, 452  
AbstractReferenceSystemType 106, 107, 451, 452  
AbstractRingPropertyType 52, 53, 401, 402  
AbstractRingType 53, 75, 401, 402, 429  
AbstractSingleOperationType 152, 153, 156, 311, 312  
AbstractSolidType 85, 86, 89, 405, 435, 436, 579, 588, 595  
AbstractStyleType 268, 269, 364  
AbstractSurfacePatchType 73, 74, 75, 77, 426, 427, 428, 429, 588  
AbstractSurfaceType 51, 52, 73, 76, 88, 399, 401, 405, 425, 426, 579, 588, 595  
AbstractTimeComplexType 185, 193, 457, 467  
AbstractTimeGeometricPrimitiveType 185, 186, 457, 458  
AbstractTimeObjectType 183, 185, 290, 455, 456, 457  
AbstractTimePrimitiveType 183, 185, 192, 456, 457, 467  
AbstractTimeReferenceSystemType 461, 462, 463, 464, 465  
AbstractTimeSliceType 204, 205, 375  
AbstractTimeTopologyPrimitiveType 192, 467, 468, 469  
AbstractTopologyType 172, 177, 178, 179, 470, 474, 475, 476  
AbstractTopoPrimitiveType 172, 173, 174, 175, 470, 471, 472, 473  
actuate 25, 488  
AestheticCriteriaType 276, 369  
AffinePlacement 70, 71, 417, 419, 507  
AffinePlacementType 70, 417  
aggregation 25, 26, 184, 442, 457, 538, 566, 567, 573  
algorithm 83, 147, 149, 150, 162, 252, 254, 308, 319, 434  
anchorPoint 137, 138, 355  
angle 8, 56, 60, 62, 64, 78, 81, 98, 123, 133, 160, 213, 218, 219, 225, 227, 277, 447  
AngleChoiceType 143, 227, 361  
AngleType 62, 225, 237, 288, 374, 415, 416, 447, 492, 572, 597  
animate 366, 542  
animateColor 366, 542  
animateMotion 366, 542  
application domain 12, 215, 286, 287, 585  
application schema 1, 3, 12, 19, 20, 21, 23, 24, 25, 26, 27, 28, 30, 35, 37, 38, 39, 56, 74, 78, 87, 189, 195, 203, 211, 231, 236, 251, 266, 267, 268, 270, 271, 272, 277, 278, 280, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 301, 302, 363, 364, 372, 373, 377, 437, 442, 482, 490, 522, 529, 530, 533, 562, 568, 569, 581, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 598, 599  
Arc 57, 58, 59, 60, 61, 62, 63, 226, 410, 411, 412, 414, 415, 416, 448, 449  
ArcByBulge 61, 414  
ArcByBulgeType 61, 414  
ArcByCenterPoint 62, 63, 64, 415, 416, 554  
ArcByCenterPointType 62, 63, 415, 416  
ArcMinutesType 226, 448  
arcrole 25, 488  
ArcSecondsType 226, 449  
ArcString 57, 58, 59, 60, 61, 410, 411, 412, 414  
ArcStringByBulge 59, 60, 61, 412, 414  
ArcStringByBulgeType 59, 60, 61, 412, 414  
ArcStringType 57, 58, 410, 411  
ArcType 58, 59, 411, 412  
AreaType 224, 492, 572, 582, 597

Array 29, 91, 92, 93, 95, 97, 181, 233, 285, 286, 378, 440, 442, 477, 607  
 Array XE "Array" AssociationType 29, 286  
 ArrayAssociationType 29, 181, 285, 378, 442, 477  
 ArrayType 29, 440  
 association 3, 6, 7, 15, 21, 22, 24, 25, 26, 29, 40, 41, 99, 100, 109, 113, 130, 131, 148, 158, 162, 198, 210, 211, 286, 314, 319, 326, 339, 340, 378, 441, 442, 463, 469, 471, 472, 473, 474, 475, 476, 477, 493, 511, 512, 513, 514, 517, 520, 521, 523, 525, 530, 535, 539, 540, 545, 566, 567, 570, 572, 573, 574, 575, 576, 577, 578, 579, 594, 597, 598, 599, 607  
 AssociationAttributeGroup 24, 25, 26, 27, 30, 34, 35, 43, 48, 49, 51, 54, 85, 89, 91, 92, 93, 94, 95, 96, 97, 107, 108, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 126, 128, 129, 130, 131, 132, 133, 134, 139, 141, 143, 144, 145, 146, 152, 153, 154, 155, 156, 157, 158, 159, 164, 165, 166, 168, 173, 174, 175, 176, 177, 180, 181, 182, 184, 186, 187, 192, 194, 201, 209, 232, 236, 240, 249, 251, 267, 268, 269, 271, 272, 273, 274, 276, 286, 288, 310, 311, 312, 313, 314, 315, 316, 320, 321, 323, 324, 326, 327, 328, 329, 330, 331, 332, 333, 334, 337, 338, 339, 340, 341, 342, 343, 345, 346, 351, 356, 357, 358, 359, 360, 361, 362, 364, 365, 366, 367, 368, 369, 372, 374, 378, 379, 381, 382, 383, 384, 385, 386, 387, 390, 392, 394, 395, 399, 400, 403, 404, 405, 406, 436, 442, 443, 444, 450, 452, 453, 456, 458, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 477, 485, 574, 577, 582, 583, 594  
 AssociationType 24, 26, 27, 35, 39, 240, 285, 369, 378, 441, 450, 589, 590, 591  
 attribute 3, 5, 7, 15, 18, 19, 20, 21, 23, 24, 25, 28, 29, 30, 31, 32, 33, 35, 40, 41, 42, 43, 44, 45, 46, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 72, 74, 75, 76, 77, 79, 80, 87, 104, 106, 118, 125, 126, 127, 137, 138, 140, 142, 144, 145, 146, 150, 153, 154, 162, 164, 165, 167, 172, 173, 174, 175, 176, 180, 181, 182, 184, 185, 188, 190, 204, 208, 210, 211, 212, 213, 214, 215, 216, 224, 226, 227, 230, 231, 234, 238, 242, 243, 245, 248, 249, 250, 252, 255, 256, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 284, 285, 301, 492, 493, 495, 497, 502, 503, 504, 506, 508, 509, 514, 516, 519, 521, 522, 523, 525, 526, 527, 528, 531, 532, 536, 539, 541, 543, 544, 566, 570, 572, 575, 576, 578, 579, 586, 587, 589, 594, 602, 603, 604, 605, 606  
 axisAbbrev 43, 125, 126, 336, 392  
 axisDirection 125, 126, 336  
 axisID 125, 335  
 axisName 125, 245, 246, 256, 264, 265, 335

**B**

Bag 29, 440  
 BagType 29, 440  
 bandwidth 278  
 baseCRS 112, 329  
 baseCurve 68, 69, 407, 408, 498, 507  
 BaseStyleDescriptorType 271, 272, 273, 276, 366, 367, 368, 369, 607  
 baseSurface 76, 77, 425, 426, 498  
 BaseUnit 214, 217, 218, 479  
 BaseUnitType 214, 479  
 basicTypes.xsd 14, 211, 224, 228, 296, 303, 390, 439, 445, 446, 547, 601  
 Bezier 67, 68, 424  
 BezierType 67, 424  
 boolean 18, 19, 20, 230, 572  
 Boolean 229, 230, 236, 292, 486  
 booleanList 19  
 BooleanList 229, 230  
 booleanOrNull 18, 19, 20, 230, 304, 482, 572  
 booleanOrNullList 19, 230, 482, 572  
 BooleanPropertyType 236  
 booleanValue 160, 317  
 boundary 3, 4, 5, 6, 7, 38, 42, 52, 53, 73, 74, 75, 76, 81, 84, 86, 87, 88, 89, 171, 172, 173, 174, 175, 176, 179, 191, 192, 401, 402, 405, 425, 427, 428, 432, 434, 436, 471, 472, 473, 493, 496, 501, 509, 520, 553  
 boundedBy 32, 33, 39, 40, 41, 239, 247, 258, 259, 261, 262, 263, 264, 348, 349, 350, 351, 377, 379, 544  
 BoundedFeatureType 39, 247  
 boundingBox 105, 454  
 boundingPolygon 105, 106, 454  
 BoundingShapeType 33, 377  
 Box i, 33, 41  
 BSpline 65, 66, 67, 422, 424, 506  
 BSplineType 65, 67, 422, 424  
 by reference 15, 24, 25, 26, 27, 187, 193, 209, 238, 285, 372, 442  
 by value 24, 25, 193, 442, 574, 575

## C

CalDate 189, 198, 461, 525  
calendar xxii, 185, 188, 189, 195, 197, 198, 199, 457, 460, 464, 465, 525, 528  
CartesianCS 116, 129, 329, 339  
cartesianCSRef 129  
CartesianCSRefType 116, 129, 329, 339  
CartesianCSType 129, 339  
catalog 214, 480  
catalogSymbol 43, 213, 217, 218, 219, 220, 221, 222, 223, 392, 479  
category 3, 4, 8, 117, 136, 228, 331, 571  
Category 230, 233, 235, 236, 250, 292, 293, 485, 486  
CategoryExtent 235, 485  
CategoryExtentType 235, 485  
CategoryList 230, 250, 292  
Category.PropertyType 236  
centerLineOf 11, 37, 268  
centerOf 37  
child 3, 7, 23, 24, 286, 287, 292, 294, 295, 301, 463  
Circle 59, 63, 412, 416  
CircleByCenterPoint 63, 64, 416, 554  
CircleByCenterPointType 63, 416  
CircleType 59, 412  
class 3, 8, 10, 13, 22, 25, 35, 41, 103, 110, 111, 124, 149, 171, 203, 204, 210, 228, 280, 301, 302, 325, 439, 442, 451, 491, 493, 495, 496, 498, 501, 507, 511, 512, 514, 516, 520, 522, 524, 527, 530, 531, 532, 536, 537, 538, 539, 540, 541, 543, 544, 545, 546, 547, 549, 562, 563, 564, 565, 566, 567, 569, 570, 571, 572, 573, 574, 575, 576, 578, 579, 586, 593, 595, 597, 598, 599  
closure 4, 5  
Clothoid 70, 71, 418, 507  
ClothoidType 70, 71, 418  
coboundary 171, 172, 173, 174, 175, 176, 472, 473  
code 18, 20, 28, 34, 104, 118, 126, 138, 144, 145, 163, 169, 210, 211, 213, 214, 217, 218, 219, 220, 221, 222, 228, 230, 233, 234, 235, 236, 238, 262, 292, 492, 571, 597  
code space 104, 453  
codelist 4, 230, 482  
CodeListType 20  
CodeOrNullListType 20, 21, 228, 230, 235, 292, 482, 485  
codespace 4  
codeSpace 18, 20, 28, 34, 104, 118, 126, 138, 144, 145, 210, 211, 214, 217, 218, 219, 220, 221, 222, 230, 233, 234, 235, 238, 262  
CodeType 18, 28, 34, 104, 118, 126, 138, 144, 145, 163, 169, 213, 214, 228, 230, 236, 292, 320, 331, 336, 352, 355, 357, 374, 379, 443, 453, 480, 482, 492, 571, 597  
columnIndex 170, 171, 353  
CompassPoint 206, 236, 237, 238, 241, 242, 374  
CompassPointEnumeration 237, 374  
composite curve 4, 76, 87, 88, 404, 428, 429, 496  
composite solid 4, 89, 405  
composite surface 4, 86, 88, 404, 405, 436, 496  
CompositeCurve 36, 76, 87, 88, 89, 404, 406, 496, 498, 499, 513, 579, 588, 595  
CompositeCurveType 88, 404, 579, 588, 595  
CompositeSolid 36, 87, 89, 405, 406, 579, 588, 595  
CompositeSolidType 89, 405, 579, 588, 595  
CompositeSurface 36, 87, 88, 89, 404, 405, 406, 513, 579, 588, 595  
CompositeSurfaceType 88, 404, 579, 588, 595  
CompositeValue 232, 233, 234, 251, 252, 253, 256, 260, 262, 264, 265, 483, 484  
CompositeValueType 232, 234, 483, 484  
composition 25, 158, 159, 162, 228, 229, 314, 316, 318, 319, 442, 530, 567, 573, 599  
CompoundCRS 113, 326  
compoundCRSRef 113  
CompoundCRSRefType 113, 326  
CompoundCRSType 113, 326

ConcatenatedOperation 155, 156, 310  
concatenatedOperationRef 156  
ConcatenatedOperationRefType 156, 310  
ConcatenatedOperationType 155, 310  
conceptual 1, 3, 12, 17, 24, 31, 73, 171, 182, 183, 242, 243, 425, 490, 492, 530, 533, 541, 544, 547, 548, 560  
conceptual schema language 12, 17, 547  
Cone 79, 80, 430, 509  
ConeType 79, 80, 430  
conformance 1, 2, 11, 12, 168, 280, 287, 294, 297, 298, 299, 301, 302, 307, 325, 335, 351, 354, 363, 443, 445, 451, 459, 478, 513, 514, 520, 521, 529, 533, 542, 543, 547, 562, 586  
container 172, 177, 470, 471  
ContainerPropertyType 177, 471  
control point 4, 50, 53, 54, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 72, 79, 80, 81, 84, 135, 168, 395, 402, 409, 410, 411, 412, 413, 414, 415, 416, 420, 421, 422, 423, 424, 425, 430, 431, 432, 435, 493, 552  
convenience 17, 23, 25, 28, 31, 33, 41, 98, 171, 184, 203, 213, 224, 227, 231, 232, 236, 282  
ConventionalUnit 214, 215, 221, 223, 479  
ConventionalUnitType 215, 479  
conversion 5, 101, 102, 110, 112, 117, 147, 148, 151, 153, 154, 157, 158, 215, 216, 218, 222, 309, 313, 314, 329, 330, 331, 479, 480, 530, 532, 569, 592  
Conversion 157, 158, 216, 313, 314, 480  
conversionRef 158  
ConversionRefType 158, 314  
conversionToPreferredUnit 215, 216, 221, 223, 479  
ConversionToPreferredUnitType 216, 480  
ConversionType 157, 313  
coord 45, 46, 47, 48, 50, 53, 57, 58, 60, 61, 62, 64, 65, 67, 111, 112, 114, 115, 116, 119, 120, 121, 126, 128, 129, 130, 131, 132, 133, 134, 150, 151, 152, 153, 154, 156, 308, 310, 311, 313, 315, 327, 328, 329, 330, 332, 333, 334, 338, 339, 340, 341, 342, 343, 393, 396, 398, 403, 410, 411, 412, 413, 414, 415, 421, 423, 424  
coordinate xxi, 4, 5, 6, 7, 33, 41, 43, 44, 45, 46, 47, 48, 50, 53, 54, 55, 57, 58, 59, 60, 61, 62, 63, 64, 66, 68, 72, 87, 97, 98, 99, 100, 101, 102, 103, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 139, 143, 144, 146, 147, 148, 149, 150, 151, 152, 153, 155, 156, 157, 158, 162, 168, 169, 188, 195, 196, 202, 207, 242, 245, 249, 251, 277, 281, 291, 292, 294, 295, 307, 308, 309, 310, 311, 312, 313, 314, 315, 319, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 351, 352, 354, 356, 358, 359, 391, 392, 393, 395, 396, 397, 398, 402, 407, 409, 410, 411, 413, 414, 421, 423, 424, 445, 451, 452, 453, 460, 462, 465, 492, 502, 528, 531, 532, 545  
coordinate reference system xxi, 4, 5, 7, 33, 41, 43, 44, 45, 46, 47, 50, 54, 57, 58, 59, 60, 61, 64, 66, 68, 72, 97, 98, 99, 100, 101, 102, 103, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 126, 127, 134, 135, 136, 137, 143, 144, 146, 147, 148, 150, 151, 153, 155, 158, 168, 207, 249, 277, 281, 291, 292, 294, 295, 308, 309, 310, 312, 315, 316, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 336, 337, 354, 356, 358, 391, 395, 396, 397, 402, 410, 411, 413, 414, 421, 423, 424, 445, 452, 453, 531, 545  
coordinate system xxi, 4, 5, 6, 55, 60, 62, 87, 107, 108, 109, 111, 114, 115, 116, 118, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 137, 139, 148, 149, 168, 188, 195, 196, 202, 325, 327, 328, 329, 331, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 354, 359, 407, 413, 414, 452, 462, 465, 492, 528  
coordinate tuple 4, 46, 47, 48, 50, 53, 57, 58, 59, 109, 115, 121, 122, 123, 126, 127, 147, 148, 149, 156, 157, 245, 251, 311, 327, 336, 337, 392, 393, 395, 398, 402, 409  
coordinateOperationID 150, 151, 153, 154, 308, 313, 315  
coordinateOperationName 150, 151, 153, 154, 308, 313, 315  
coordinateOperationRef 152, 156, 310, 311  
CoordinateOperationRefType 152, 156, 309, 310, 311  
coordinateOperations.xsd 98, 103, 111, 149, 307, 325, 451, 601  
coordinateReferenceSystemRef 111, 112, 114, 115, 116, 119, 120, 121, 327, 328, 329, 330, 332, 333, 334  
CoordinateReferenceSystemRefType 111, 112, 113, 114, 115, 116, 119, 120, 121, 325, 326, 327, 328, 329, 330, 332, 333, 334  
coordinateReferenceSystems.xsd 42, 97, 103, 110, 150, 278, 324, 325, 391, 438, 451, 601  
coordinates 45, 47, 48, 50, 53, 57, 58, 60, 61, 62, 64, 65, 67, 393, 396, 398, 403, 410, 411, 412, 413, 414, 415, 421, 423, 424  
CoordinatesType 45, 46, 251, 346, 398, 572  
CoordinateSystemAxis 125, 126, 128, 335, 336, 338  
CoordinateSystemAxisBaseType 125, 335  
coordinateSystemAxisRef 126  
CoordinateSystemAxisRefType 126, 128, 336, 338  
CoordinateSystemAxisType 125, 335  
coordinateSystemRef 128, 129, 130, 131, 132, 133, 134, 338, 339, 340, 341, 342, 343  
CoordinateSystemRefType 118, 128, 129, 130, 131, 132, 133, 134, 331, 338, 339, 340, 341, 342, 343

coordinateSystems.xsd 98, 111, 124, 325, 334, 601  
CoordType 46, 398  
Count 230, 235, 236, 292, 293, 485, 487  
CountExtent 235, 236, 485  
CountExtentType 235, 485  
CountList 230, 293  
Count.PropertyType 236  
covariance 168, 170, 171, 353  
CovarianceElementType 170, 353  
covarianceMatrix 170  
CovarianceMatrixType 170, 353  
coverage xxi, 5, 12, 14, 212, 242, 243, 244, 247, 248, 249, 250, 252, 254, 255, 257, 259, 260, 261, 262, 263, 264, 278, 292, 293, 343, 344, 345, 347, 348, 349, 350, 438, 545, 547, 561, 601  
coverage.xsd 244, 247, 278, 292, 343, 438, 601  
coverageFunction 248, 249, 254, 258, 259, 261, 262, 263, 264, 344, 345, 348, 349, 350, 351  
CoverageFunctionType 254, 345  
CRS 9, 24, 42, 43, 44, 45, 47, 99, 100, 101, 103, 104, 105, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 118, 119, 120, 121, 122, 124, 126, 134, 135, 136, 138, 145, 147, 148, 150, 151, 163, 282, 290, 308, 309, 320, 325, 327, 328, 329, 330, 331, 332, 333, 334, 336, 355, 357, 391, 392, 396, 397, 451, 452, 453, 454, 495, 531, 532, 580  
crsRef 108, 111, 113, 325, 326  
CRSRefType 108, 111, 113, 151, 309, 325, 326, 453  
csID 127, 128, 337  
csName 127, 337  
CSV 9, 250, 307  
CT 9  
CubicSpline 64, 65, 421, 493  
CubicSplineType 64, 421  
curve 4, 5, 6, 11, 44, 45, 49, 50, 55, 56, 57, 58, 59, 60, 61, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 76, 77, 78, 79, 81, 87, 88, 93, 126, 173, 176, 178, 186, 259, 336, 383, 391, 394, 395, 404, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 428, 429, 430, 432, 472, 474, 496, 504, 505, 506, 511, 513, 552, 554, 579  
Curve 36, 37, 49, 50, 55, 56, 57, 58, 60, 62, 64, 65, 67, 68, 69, 72, 76, 79, 80, 88, 93, 379, 388, 394, 395, 407, 409, 410, 411, 413, 416, 420, 422, 423, 425, 428, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599  
CurveInterpolationType 56, 57, 58, 60, 62, 64, 65, 67, 72, 79, 80, 410, 411, 413, 416, 420, 422, 423, 425, 431, 432  
curveMember 75, 76, 88, 92, 93, 260, 383, 404, 429  
curveMembers 92, 93, 383  
curveProperty 49, 173, 394, 472, 607  
curveProperty, Curve.PropertyType 36, 37, 49, 51, 68, 69, 76, 173, 517, 578, 579, 607  
Curve.PropertyType 37, 49, 68, 69, 76, 379, 394, 407, 416, 428, 578, 579  
CurveSegment 55, 56, 57, 59, 62, 64, 65, 69, 70, 71, 575, 583, 588  
CurveSegmentArray.PropertyType 56, 409  
CurveType 55, 407, 579, 588, 595  
Cylinder 80, 431, 509  
CylinderType 80, 431  
CylindricalCS 133, 342, 343  
cylindricalCSRef 133  
CylindricalCSRefType 133, 342  
CylindricalCSType 133, 342

## D

dangling 174, 175, 176, 472, 473  
data type 5, 8, 11, 13, 16, 44, 45, 282, 396, 446, 461, 524, 529, 555, 564, 565, 566, 567, 572, 576, 580, 587, 590, 591, 598, 599  
DataBlock 250, 251, 252, 256, 260, 265, 345, 346  
DataBlockType 250, 346  
dataQuality.xsd 98, 150, 168, 308, 351, 601  
dataSource 203, 204, 375, 376

datum xxi, 4, 5, 6, 99, 107, 108, 111, 112, 114, 115, 119, 120, 121, 126, 134, 135, 136, 137, 138, 139, 140, 143, 144, 145, 146, 153, 158, 313, 315, 325, 327, 328, 329, 332, 333, 334, 336, 354, 355, 356, 357, 358, 359, 360, 452  
datumID 137, 138, 146, 355, 358  
datumName 137, 138, 146, 354, 358  
datumRef 138, 139, 143, 144, 145, 146, 356, 357, 358, 359  
DatumRefType 138, 139, 143, 144, 145, 146, 355, 356, 357, 358, 359, 360  
datums.xsd 98, 111, 136, 325, 354, 601  
decimalMinutes 225, 226, 227, 448  
DecimalMinutesType 226, 448  
defaultStyle 267, 268, 535, 536, 541, 543  
defaultStyle.xsd 14, 266, 278, 363, 438, 601  
DefaultStylePropertyType 267, 363  
definedByConversion 112, 329  
definition 209, 210, 211  
Definition 3, 9, 106, 125, 127, 137, 140, 141, 142, 150, 162, 164, 195, 196, 198, 200, 207, 208, 209, 210, 211, 212, 213, 217, 218, 222, 223, 280, 293, 295, 308, 319, 321, 335, 337, 354, 360, 361, 371, 372, 373, 451, 462, 463, 464, 478, 479  
DefinitionCollection 207, 208, 210, 211, 217, 218, 222, 223  
definitionMember 208, 209, 210, 211  
DefinitionProxy 209, 210, 211, 373  
DefinitionProxyType 209, 373  
definitionRef 209, 210, 211, 373  
DefinitionType 106, 125, 127, 137, 140, 142, 150, 151, 162, 163, 164, 165, 167, 195, 196, 198, 200, 207, 208, 209, 213, 295, 308, 319, 321, 335, 337, 354, 360, 361, 371, 372, 373, 451, 462, 463, 464, 479  
degrees 56, 78, 81, 98, 99, 108, 168, 218, 225, 226, 227, 447  
DegreesType 226, 448  
DegreeValueType 226, 448  
deprecate xx, 9  
derivation 23, 25, 39, 147, 155, 158, 208, 222, 257, 270, 281, 284, 285, 296, 315, 316, 499, 512, 525, 530, 578, 606  
derivationUnitTerm 214, 215, 218, 219, 220, 221, 222, 223, 479  
DerivationUnitTermType 215, 480  
DerivedCRS 117, 118, 331, 332  
derivedCRSRef 118  
DerivedCRSRefType 118, 331  
derivedCRSType 117, 331  
DerivedCRSType 117, 331  
DerivedCRSTypeType 117, 331  
DerivedUnit 214, 215, 218, 219, 220, 221, 222, 223, 479  
DerivedUnitType 214, 479  
description 1, 8, 11, 14, 15, 24, 25, 26, 28, 39, 42, 46, 99, 104, 105, 121, 122, 145, 169, 182, 183, 185, 186, 188, 196, 197, 198, 199, 201, 207, 208, 210, 211, 212, 213, 217, 218, 222, 223, 224, 239, 242, 243, 244, 250, 251, 254, 265, 267, 274, 283, 295, 302, 371, 439, 454, 491, 495, 516, 522, 527, 529, 533, 537, 539, 540, 569, 592  
dictionary xxii, 14, 18, 20, 104, 207, 208, 209, 210, 212, 217, 230, 295, 305, 306, 371, 372, 373, 461, 478, 482, 549, 569, 578, 601  
Dictionary 8, 107, 126, 128, 138, 141, 143, 152, 164, 207, 208, 209, 210, 211, 217, 224, 283, 295, 309, 320, 321, 336, 338, 356, 361, 362, 371, 372, 452, 549  
dictionary.xsd 104, 295, 371, 461, 478, 601  
dictionaryEntry 107, 126, 128, 138, 141, 143, 152, 164, 208, 209, 210, 211, 217, 218, 219, 220, 221, 222, 223, 224, 309, 320, 321, 336, 338, 355, 361, 362, 372, 452  
DictionaryEntryType 107, 126, 128, 138, 141, 143, 152, 164, 208, 209, 309, 320, 321, 336, 338, 356, 361, 362, 372, 452  
DictionaryType 208, 295, 371  
dimension 43, 47, 70, 87, 98, 121, 123, 171, 176, 179, 185, 245, 246, 251, 256, 260, 261, 262, 263, 265, 344, 389, 390, 391, 398, 418, 444, 502, 554, 555  
direct position 4, 5, 6, 41, 44, 45, 47, 48, 50, 54, 57, 58, 59, 60, 61, 63, 65, 70, 72, 187, 242, 248, 249, 344, 391, 395, 397, 398, 402, 410, 411, 413, 414, 415, 418, 421  
directed 6, 14, 171, 172, 173, 174, 175, 176, 178, 179, 191, 193, 276, 277, 289, 369, 466, 472, 473, 474, 475, 519, 520, 555  
directedEdge 172, 174, 178, 471, 472, 473, 474, 607  
DirectedEdgePropertyType 174, 472, 580  
directedFace 173, 175, 178, 472, 473, 475, 607  
DirectedFacePropertyType 175, 473, 580  
directedNode 173, 177, 471, 472, 474, 607  
DirectedNodePropertyType 173, 289, 290, 471, 580

DirectedObservation 239, 241, 242, 450  
 DirectedObservationAtDistance 241, 242, 450  
 DirectedObservationAtDistanceType 242, 450  
 DirectedObservationType 241, 242, 450  
 directedTopoSolid 174, 175, 176, 179, 473, 475  
 DirectedTopoSolid.PropertyType 175, 176, 473, 580  
 direction 236, 237, 238, 241, 242, 450  
 direction.xsd 47, 236, 373, 375, 449, 601  
 Direction.PropertyType 205, 236, 373, 375  
 DirectionVector 236, 237, 374  
 DirectionVectorType 237, 374  
 DirectPositionListType 44, 45, 396  
 DirectPositionType 44, 46, 47, 48, 50, 54, 57, 58, 59, 60, 61, 63, 64, 66, 68, 70, 72, 396, 397, 398, 418  
 dmsAngle 160, 225, 226, 227, 317, 447  
 DMSAngleType 160, 225, 317, 447  
 dmsAngleValue 160, 317  
 domain xxi, 3, 4, 5, 6, 8, 12, 31, 39, 48, 49, 51, 54, 85, 90, 91, 92, 93, 94, 95, 96, 97, 105, 106, 195, 242, 243, 248, 249, 250, 254, 255, 257, 258, 259,  
     260, 262, 263, 264, 283, 286, 287, 291, 344, 345, 347, 348, 349, 350, 381, 382, 383, 384, 385, 386, 387, 392, 393, 395, 399, 400, 403, 406, 435,  
     454, 455, 462, 545, 575, 578, 585  
 domainSet 247, 248, 249, 256, 258, 259, 260, 261, 262, 344, 348, 349, 350, 351  
 DomainSetType 249, 258, 263, 264, 345, 348, 349, 350, 351  
 double 18, 19, 20, 21, 44, 251, 572  
 doubleList 19, 21, 44, 306, 396, 397  
 doubleOrNull 18, 19, 20, 21, 250, 251, 252, 260, 265, 305, 307, 346, 572  
 doubleOrNullList 19, 21, 251, 307, 346, 572  
 doubleOrNullTupleList 250, 251, 252, 260, 265, 346  
 DrawingType.Type 276, 369  
 DTD 9  
 duration xxii, 38, 182, 185, 186, 187, 190, 191, 199, 217, 276, 524  
 dynamicFeature.xsd 278, 374, 438, 601  
 dynamicProperties 203, 376

## E

edge 5, 173, 174, 176, 178, 191, 192, 193, 194, 468, 469, 472, 474  
 Edge 173, 174, 176, 470, 472  
 edgeOf 37  
 EdgeType 173, 472  
 element xxi, 3, 5, 6, 7, 13, 14, 15, 17, 18, 19, 20, 21, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48,  
     49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88,  
     89, 90, 91, 92, 93, 94, 95, 96, 97, 99, 101, 102, 103, 104, 105, 106, 107, 108, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 123, 125, 126,  
     127, 128, 129, 130, 131, 132, 133, 134, 135, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159,  
     160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 189,  
     190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 203, 204, 205, 206, 207, 208, 209, 210, 212, 213, 214, 215, 216, 224, 225, 226, 227,  
     228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 239, 240, 241, 242, 244, 245, 246, 247, 248, 249, 250, 251, 252, 254, 255, 258, 259, 260, 261,  
     262, 263, 264, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 278, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295,  
     301, 492, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 507, 508, 509, 510, 511, 512, 513, 514, 516, 517, 519, 520, 522, 523, 524, 526, 527,  
     528, 529, 532, 536, 539, 540, 541, 552, 553, 562, 568, 569, 570, 573, 574, 575, 576, 577, 578, 579, 581, 582, 583, 586, 587, 589, 591, 593, 594,  
     600, 602, 603, 604, 605, 606  
 Ellipsoid 114, 128, 129, 139, 140, 141, 327, 338, 339, 359, 361, 362  
 EllipsoidalCS 114, 128, 129, 327, 338, 339  
 ellipsoidalCSRef 128  
 EllipsoidalCSRefType 114, 128, 327, 338  
 EllipsoidalCSType 128, 338  
 EllipsoidBaseType 140, 362  
 ellipsoidID 140, 362  
 ellipsoidName 140, 361  
 ellipsoidRef 141  
 EllipsoidRefType 139, 141, 359, 362

EllipsoidType 140, 361  
 encapsulate 206, 288, 289, 290, 536  
 Engineering 9, 100, 101, 109, 110, 122, 123, 134, 136  
 EngineeringCRS 118, 119, 332  
 engineeringCRSRef 119  
 EngineeringCRSRefType 119, 332  
 EngineeringCRSType 118, 332  
 EngineeringDatum 119, 143, 144, 332, 356  
 engineeringDatumRef 143  
 EngineeringDatumRefType 119, 143, 332, 356  
 EngineeringDatumType 143, 356  
 Envelope 17, 33, 41, 47, 105, 106, 283, 377, 378, 398, 455, 502  
 EnvelopeType 33, 47, 105, 106, 283, 378, 398, 455  
 EnvelopeWithTimePeriod 33, 377  
 EnvelopeWithTimePeriodType 33, 377  
 EPSG 9, 25, 34, 233, 234, 247, 258, 260, 261, 265, 532  
 error 98, 108, 111, 147, 151, 155, 158, 168, 169, 278, 280, 281, 309, 315, 316, 325, 352, 586  
 extentOf 37, 273  
 ExtentType 105, 454  
 exterior 5, 7, 52, 73, 74, 75, 76, 86, 261, 262, 401, 402, 425, 427, 428, 436, 496, 501, 509, 553, 607

**F**

face 6, 173, 174, 175, 176, 178, 472, 475  
 Face 174, 175, 177, 471, 472, 473  
 FaceType 174, 472  
 factor 216, 223  
 feature xxi, 1, 5, 6, 7, 8, 11, 12, 13, 14, 15, 21, 22, 24, 31, 32, 33, 35, 36, 37, 38, 39, 40, 41, 44, 99, 100, 109, 114, 122, 126, 131, 138, 171, 177, 178, 179, 182, 191, 192, 193, 195, 202, 203, 204, 205, 206, 211, 212, 236, 238, 243, 244, 247, 248, 249, 266, 267, 268, 269, 270, 271, 272, 273, 275, 277, 278, 280, 284, 286, 287, 288, 292, 293, 294, 301, 327, 336, 340, 343, 344, 355, 365, 366, 367, 368, 369, 375, 376, 377, 378, 379, 439, 449, 466, 469, 474, 475, 477, 529, 530, 536, 537, 538, 539, 540, 541, 543, 560, 563, 564, 565, 566, 573, 576, 587, 589, 591, 592, 600, 601  
 Feature 25, 28, 32, 39, 40, 41, 203, 239, 268, 270, 286, 287, 292, 301, 570, 573, 574, 581, 582, 587, 589, 593, 595  
 feature collection xxi, 11, 31, 39, 40, 41, 182, 204, 205, 211, 243, 266, 267, 268, 280, 286, 287, 293, 376, 378, 379, 477, 536, 560, 573  
 feature relationship 6, 466  
 feature.xsd 14, 31, 36, 277, 278, 284, 286, 343, 375, 376, 449, 560, 600, 601  
 FeatureArrayPropertyType 35, 378  
 FeatureCollection 39, 40, 41, 203, 204, 283, 286, 287, 376, 378, 573, 589, 591, 607  
 FeatureCollectionType 40, 203, 376, 378  
 featureMember 32, 35, 40, 204, 378, 573, 591, 607  
 featureMember XE "featureMember" s 35, 204  
 featureMembers 32, 35, 40, 378, 607  
 featureProperty 35, 600, 606, 607  
 Feature.PropertyType 35, 239, 378, 449  
 featureStyle 268, 269, 364, 541  
 FeatureStyle 269, 270, 273, 274, 275, 276, 364, 365, 536, 539, 541  
 FeatureStyle.PropertyType 269, 364  
 FeaturestyleType 269, 365  
 File 250, 252, 253, 262, 264, 345, 346  
 FileType 252, 346  
 FileValueModelType 252, 346  
 formula 71, 163, 212, 216, 221, 320, 419, 479, 480, 481  
 FormulaType 216, 481  
 from 488  
 function xxi, 5, 6, 7, 22, 55, 56, 64, 71, 77, 78, 81, 147, 243, 248, 254, 256, 259, 260, 261, 262, 269, 274, 280, 299, 344, 345, 408, 409, 418, 421, 431, 540, 543

**G**

generalConversionRef 154, 158, 314  
 GeneralConversionRefType 112, 154, 158, 313, 314, 330  
 generalDerivedCRSRef 112, 117, 118, 330, 331

GeneralDerivedCRSRefType 112, 117, 118, 330, 331, 332  
generalTransformationRef 155, 159, 316  
GeneralTransformationRefType 155, 159, 315, 316  
GenericMetaData 31, 441  
GenericMetaDataType 31, 441  
Geocentric 100, 108, 110, 122, 124, 134  
GeocentricCRS 115, 116, 328, 329  
geocentricCRSRef 116  
GeocentricCRSRefType 116, 329  
GeocentricCRSType 115, 328  
Geodesic 71, 72, 420, 507  
GeodesicString 71, 72, 420, 507  
GeodesicStringType 71, 72, 420  
GeodesicType 72, 420  
GeodeticDatum 114, 139, 327, 359, 360  
geodeticDatumRef 139  
GeodeticDatumRefType 114, 139, 327, 359  
GeodeticDatumType 139, 359  
Geographic i, 3, xx, xxi, 1, 2, 3, 9, 99, 100, 101, 105, 109, 110, 122, 124, 301, 445, 454, 478, 532, 614  
GeographicCRS 114, 326, 327, 606  
geographicCRSRef 114  
GeographicCRSRefType 114, 327  
GeographicCRSType 114, 326  
geometric aggregate xxi, 42, 86, 90, 91, 92, 93, 95, 97, 380, 381, 382, 383, 384, 385, 554  
geometric complex xxii, 86, 87, 89, 90, 184, 345, 406, 457  
geometric object 4, 6, 45, 47, 60, 62, 78, 87, 392, 413, 414, 523, 540  
geometric property 11, 35, 37, 43, 390  
geometric set 6  
GeometricAggregate 87, 90, 91, 92, 94, 95, 96  
GeometricComplex 41, 87, 89, 406, 512, 579, 596  
GeometricComplexPropertyType 89, 512, 579  
GeometricComplexType 87, 89, 406, 579, 596  
geometricPositionGroup 45, 72, 83, 398, 420, 435, 579, 583  
GeometricPositionGroup 45, 579, 583  
geometricPositionListGroup 45, 77, 429, 579  
GeometricPositionListGroup 45, 579  
GeometricPrimitive 42, 47, 48, 49, 51, 85, 87  
GeometricPrimitivePropertyType 48, 89, 406  
Geometry 33, 41, 42, 43, 44, 47, 53, 89, 90, 245, 288, 301, 579, 588, 595  
geometry model 41, 42, 54, 86, 87  
geometryAggregates.xsd 86, 344, 380, 404, 601  
GeometryArrayPropertyType 44, 91, 288, 387  
geometryBasic0d1d.xsd 41, 228, 229, 373, 389, 399, 444, 481, 601  
geometryBasic2d.xsd 31, 41, 104, 277, 377, 399, 407, 451, 601  
geometryComplexes.xsd 86, 171, 287, 404, 470, 601  
geometryMember 90, 91, 381  
geometryMembers 90, 91, 381  
geometryPrimitives.xsd 54, 380, 406, 407, 601  
geometryProperty 271  
Geometry.PropertyType 43, 44, 91, 288, 387, 579  
geometryStyle 269, 270, 365, 541  
GeometryStyle 270, 271, 273, 275, 366, 540, 541  
GeometryStylePropertyType 271, 366  
GeometryStyleType 271, 366  
GIS i, ii, xx, 9  
global 24, 26, 38, 39, 99, 172, 173, 174, 175, 267, 280, 281, 285, 287, 292, 294, 295, 344, 409, 410, 411, 412, 414, 415, 416, 418, 420, 422, 424, 443, 444, 480, 487, 491, 569, 570, 573, 587, 589, 591, 593, 594, 595, 600  
gml i, 3, i, xxi, 1, 9  
gml.xsd 278, 284, 285, 438, 601, 606

gmlBase.xsd 13, 14, 23, 25, 229, 277, 371, 390, 438, 455, 478, 548, 601  
 grammar xxi, 11, 270, 273, 274, 365, 370, 536, 543  
 graph 184, 192, 193, 195, 199, 202, 243, 269, 274, 275, 276, 277, 365, 368, 369, 466  
 graphStyle 268, 276, 364, 541  
 GraphStyle 269, 276, 368, 541  
 GraphStyle.PropertyType 276, 368  
 GraphStyleType 276, 368  
 GraphType.Type 276, 369  
 greenwichLongitude 142, 143, 360  
 grid 6, 7, 14, 79, 101, 136, 144, 149, 242, 244, 245, 246, 255, 263, 264, 276, 345, 347, 357, 369, 429, 430, 444, 445, 446, 545  
 Grid 244, 245, 246, 254, 255, 256, 263, 264, 265, 346, 347, 350, 444, 445  
 GridCoverage 263, 350  
 GridCoverageType 263, 350  
 GriddedSurface 78, 79, 80  
 gridDomain 263, 264, 350  
 GridDomainType 350  
 GridEnvelopeType 245, 445  
 GridFunction 254, 255, 346, 347  
 GridFunctionType 255, 347  
 GridLengthType 224  
 GridLimitsType 245, 444  
 grids.xsd 47, 244, 263, 264, 265, 344, 444, 601  
 GridType 244, 246, 444, 445  
 groupID 166, 167, 323  
 groupName 166, 167, 323

**H**

hierarchy 12, 13, 183, 201, 207, 228, 281, 439, 460, 463, 495, 516, 522, 527, 570  
 history 189, 191, 203, 204, 205, 206, 375, 376, 466  
 History.PropertyType 206, 375, 376  
 href 7, 15, 24, 25, 26, 27, 34, 187, 196, 197, 201, 210, 211, 217, 218, 234, 238, 240, 241, 242, 249, 254, 275, 438, 488, 536, 601, 606  
 HTTP 9

**I**

id 3, 23, 24, 25, 26, 27, 28, 29, 30, 34, 39, 42, 106, 125, 127, 137, 140, 142, 146, 150, 153, 154, 162, 165, 167, 172, 186, 187, 196, 197, 201, 208, 209,  
     210, 211, 213, 217, 218, 219, 220, 221, 222, 223, 224, 244, 247, 265, 283, 301, 308, 313, 315, 319, 322, 323, 335, 337, 354, 358, 360, 361, 371,  
     372, 391, 439, 441, 451, 495, 516, 522, 527, 529, 536, 540, 587, 590, 592  
 IdentifierType 104, 107, 125, 128, 138, 140, 143, 151, 163, 166, 167, 309, 320, 322, 324, 336, 338, 355, 361, 362, 452  
 IETF 2, 9  
 Image 101, 109, 119, 122, 123, 136, 333  
 ImageCRS 119, 120, 332, 333  
 imageCRSRef 120  
 ImageCRSRefType 120, 333  
 ImageCRSType 119, 332  
 ImageDatum 120, 144, 333, 356, 357  
 imageDatumRef 144  
 ImageDatumRefType 120, 144, 333, 357  
 ImageDatumType 144, 356  
 includesCRS 113, 326  
 includesElement 170, 353  
 includesParameter 166, 167, 323  
 includesValue 161, 162, 318  
 IncrementOrder 255, 347  
 IndexMap 347  
 IndexMapType 347  
 indirectEntry 208, 209, 210, 211, 372  
 IndirectEntryType 209, 372  
 informative 278, 490, 491, 562, 581, 600  
 inheritance 87, 266, 267, 498, 512, 530, 565

innerBoundaryIs 53  
integer 18, 19, 20, 161, 230, 235, 245, 255, 293, 572  
integerList 19, 161, 245, 255, 318, 347, 445  
integerOrNull 18, 19, 20, 230, 235, 305, 483, 485, 572  
integerOrNullList 19, 230, 235, 483, 485, 572  
integerValue 160, 161, 317  
integerValueList 160, 161, 317  
interior 4, 6, 7, 52, 53, 56, 73, 74, 76, 83, 86, 99, 176, 401, 402, 409, 425, 427, 428, 434, 436, 437, 472, 496, 501, 509, 553, 607  
interoperability 280, 288, 289, 290, 291  
inverseFlattening 141, 362  
isolated 172, 176, 470  
IsolatedPropertyType 176, 470  
isSphere 141, 142, 362

## K

Knot.PropertyType 65, 66, 67, 423, 424  
Knot.Type 65, 66, 67, 422, 424, 425, 506  
KnotTypes.Type 65, 67, 424, 425

## L

label 488, 489  
labelStyle 270, 271, 272, 365, 366, 367, 540, 542  
LabelStyle 270, 272, 273, 274, 367, 368, 540, 541, 542  
LabelStyle.PropertyType 272, 367  
LabelStyle.Type 273, 368  
LabelType 273, 368  
lang 217, 307, 324, 334, 343, 351, 354, 374, 375, 376, 444, 445, 451, 455, 456, 457, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 478  
legacy system 288, 289, 290  
LengthType 62, 69, 83, 141, 224, 362, 363, 415, 416, 434, 492, 571, 573, 597  
lexical 22, 188, 189, 213, 307, 460, 461, 524, 525, 526  
line string 6, 50, 51, 56, 64, 84, 93, 94, 254, 386, 389, 395, 399, 408, 409, 421, 434, 553  
LinearCS 131, 340, 341  
linearCSRef 131  
LinearCSRef.Type 131, 340  
LinearCSType 131, 340  
LinearRing 53, 54, 75, 261, 262, 402, 403, 553, 607  
LinearRing.PropertyType 54  
LinearRingType 53, 402  
LineString 36, 45, 50, 51, 56, 57, 83, 84, 93, 260, 389, 395, 399, 409, 434, 435, 493, 504, 552, 588, 607  
lineStringMember 93, 94, 386  
lineStringProperty 51, 600, 606  
LineString.PropertyType 51, 93, 389, 399  
LineStringSegment 50, 56, 57, 83, 84, 409, 434, 435, 504  
LineStringSegmentArray.PropertyType 83, 84, 434  
LineStringSegment.Type 56, 57, 409  
LineStringType 50, 395, 588  
LineType.Type 276, 369  
list types 19, 20, 21, 43, 44, 161, 228, 230, 235, 245, 251, 255, 292, 293, 572  
local 41, 69, 70, 78, 84, 99, 100, 109, 118, 123, 136, 144, 195, 199, 207, 251, 280, 285, 307, 332, 356, 417, 434, 466, 570, 576, 594, 601, 602  
location xxi, 8, 11, 16, 23, 25, 26, 31, 32, 33, 34, 35, 37, 39, 40, 41, 42, 55, 70, 86, 104, 111, 124, 137, 139, 149, 150, 169, 182, 186, 191, 195, 202,  
204, 205, 206, 207, 212, 224, 228, 236, 239, 240, 241, 242, 243, 244, 247, 256, 268, 280, 283, 375, 377, 379, 380, 533  
LocationKeyword 33, 533, 580  
LocationKeyWord 34, 379  
Location.PropertyType 33, 34, 35, 379, 380  
LocationString 33, 34, 241, 379, 533, 580

**M**

MappingRule 254, 346  
maximalComplex 179, 476  
maximumOccurs 166, 167, 323  
measurand 7  
measure xxi, 6, 7, 12, 14, 19, 21, 43, 98, 101, 108, 110, 121, 122, 123, 126, 129, 147, 160, 161, 169, 170, 207, 211, 212, 213, 214, 215, 216, 217, 222, 224, 225, 231, 232, 275, 284, 293, 295, 296, 317, 318, 336, 339, 352, 353, 392, 445, 446, 447, 478, 479, 480, 481, 482, 483, 492  
Measure 19, 62, 69, 83, 141, 160, 170, 205, 211, 224, 225, 228, 231, 237, 242, 288, 292, 296, 492, 571, 572, 573, 582, 597  
measureDescription 169, 352  
MeasureListType 21, 161, 211, 228, 317  
MeasureOrNullListType 21, 211, 231, 235, 251, 292, 293, 483, 484  
measures.xsd 212, 238, 275, 293, 296, 363, 390, 445, 601  
MeasureType 19, 160, 170, 205, 211, 224, 225, 228, 231, 242, 292, 296, 317, 352, 375, 446, 447, 450, 483, 492, 572, 597  
member i, xx, 16, 17, 24, 27, 29, 39, 40, 76, 87, 88, 89, 167, 184, 190, 192, 193, 199, 200, 201, 202, 231, 233, 234, 236, 240, 250, 284, 285, 440, 529, 607  
members 6, 20, 29, 40, 41, 87, 91, 92, 93, 95, 97, 167, 180, 182, 199, 204, 205, 208, 211, 228, 232, 234, 245, 254, 258, 259, 260, 261, 262, 280, 286, 293, 440, 570, 604, 607  
meridianID 142, 143, 360  
meridianName 142, 360  
Metadata 24, 28, 30, 31, 39, 103, 104, 106, 125, 127, 137, 140, 142, 146, 150, 153, 154, 162, 165, 166, 183, 207, 208, 568, 592, 607  
metaDataProperty 24, 28, 30, 39, 103, 104, 106, 125, 127, 137, 140, 142, 146, 150, 153, 154, 162, 165, 166, 183, 207, 208, 308, 313, 315, 319, 322, 323, 335, 337, 354, 358, 360, 361, 371, 439, 451, 568, 592, 607  
MetaDataTableType 28, 30, 442  
method 4, 16, 24, 25, 55, 74, 75, 147, 149, 150, 153, 154, 155, 157, 158, 159, 160, 161, 162, 163, 164, 165, 167, 188, 191, 205, 225, 242, 243, 250, 252, 291, 308, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 407, 427, 428, 479, 532, 537, 540, 543, 601, 606, 609  
methodFormula 162, 163, 320  
methodID 162, 163, 319  
methodName 162, 163, 319  
minimumOccurs 164, 165, 166, 321, 322, 323  
minutes 189, 191, 225, 226, 227, 448  
model driven 12  
modelling xxi, 1, 14, 149, 171, 532  
modifiedCoordinate 156, 157, 311  
modularisation 277  
MovingObjectStatus 205, 206, 375, 376  
MovingObjectStatusType 205, 375  
multiCenterLineOf 37  
multiCenterOf 37  
multiCoverage 37  
MultiCurve 36, 37, 92, 93, 94, 259, 260, 348, 349, 383, 388, 511, 580, 596  
MultiCurveCoverage 348  
MultiCurveCoverageType 348  
multiCurveDomain 259, 349  
MultiCurveDomainType 349  
multiCurveProperty 36, 93, 383  
MultiCurve.PropertyType 36, 37, 93, 94, 383, 388, 580  
MultiCurveType 92, 93, 94, 383, 580, 596  
multiEdgeOf 37  
multiExtentOf 37  
MultiGeometry 36, 41, 90, 91, 380, 381, 554, 579, 580, 596  
multiGeometryProperty 36, 91, 381, 382  
MultiGeometry.PropertyType 36, 91, 381, 579, 580  
MultiGeometryType 90, 380, 579, 580, 596  
MultiLineString 94, 386  
MultiLineString.PropertyType 94  
MultiLineStringType 94, 386  
*multiLocation* 37  
MultiPoint 36, 37, 91, 92, 257, 258, 259, 263, 348, 381, 382, 388, 389, 511, 580, 596  
MultiPointCoverage 348

MultiPointCoverageType 348  
multiPointDomain 258, 348  
MultiPointDomainType 348  
multiPointProperty 36, 92  
MultiPoint.PropertyType 36, 37, 92, 382, 388, 389, 580  
MultiPointType 91, 92, 381, 580, 596  
MultiPolygon 95, 96, 386, 387  
MultiPolygon.PropertyType 96  
MultiPolygonType 95, 96, 386  
multiPosition 37  
MultiSolid 36, 96, 97, 262, 350, 385, 512, 580, 596  
MultiSolidCoverage 350  
MultiSolidCoverageType 350  
multiSolidDomain 262, 350  
MultiSolidDomainType 350  
multiSolidProperty 36, 97, 385  
MultiSolid.PropertyType 36, 97, 385, 580  
MultiSolidType 96, 385, 580, 596  
MultiSurface 36, 37, 94, 95, 96, 260, 261, 262, 349, 384, 388, 511, 580, 596  
MultiSurfaceCoverage 349  
MultiSurfaceCoverageType 349  
multiSurfaceDomain 261, 349  
MultiSurfaceDomainType 349  
multiSurfaceProperty 36, 95, 384, 607  
MultiSurface.PropertyType 36, 37, 95, 96, 384, 388, 580  
MultiSurfaceType 94, 96, 384, 580, 596

## N

name ii, 4, 5, 6, 8, 11, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 104, 105, 106, 107, 108, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 239, 240, 241, 242, 244, 245, 246, 247, 248, 249, 250, 251, 252, 254, 255, 258, 259, 260, 262, 263, 264, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 281, 282, 283, 287, 288, 290, 308, 319, 322, 323, 335, 337, 354, 360, 361, 371, 439, 451, 453, 492, 493, 495, 502, 504, 505, 506, 509, 512, 513, 516, 519, 520, 522, 523, 525, 526, 527, 528, 529, 530, 538, 539, 540, 544, 547, 549, 563, 564, 566, 570, 571, 573, 574, 575, 576, 577, 578, 581, 582, 583, 587, 589, 590, 591, 593, 595, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613  
Name 19, 20  
NameList 19, 20, 306  
nameOrNull 18, 20, 572  
NameOrNull 18, 19, 20, 304, 306, 572  
NameOrNull XE "NameOrNull" List 20, 572  
NameOrNullList 19, 20, 306  
NCName 19, 43  
NCNameList 19, 43, 392  
nested 69, 76, 77, 88, 89, 149, 207, 404, 405, 408, 429, 463, 587  
node 7, 172, 173, 176, 177, 191, 192, 193, 194, 468, 471, 472, 474, 539  
Node 172, 173, 176, 470, 471  
NodeType 172, 471  
normative 1, 3, 11, 13, 15, 286, 297, 301, 303, 442, 562, 585, 593  
Null 16, 17, 18, 33, 34, 229, 232, 233, 234, 236, 304, 305, 377, 379, 482, 572  
NullEnumeration 16, 18, 304, 305  
NullType 16, 17, 21, 304, 572

**O**

object 4, 6, 7, 8, 13, 15, 21, 22, 23, 24, 26, 27, 28, 29, 30, 31, 36, 37, 38, 44, 45, 47, 60, 62, 71, 72, 76, 78, 86, 104, 105, 106, 110, 123, 124, 182, 183, 184, 186, 187, 191, 192, 194, 196, 204, 205, 206, 208, 209, 211, 239, 240, 244, 248, 249, 267, 268, 269, 270, 274, 285, 286, 291, 294, 299, 301, 303, 344, 372, 373, 376, 396, 419, 420, 428, 436, 438, 439, 441, 443, 449, 450, 453, 454, 455, 491, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 516, 517, 519, 520, 521, 522, 523, 526, 527, 528, 529, 530, 532, 533, 539, 540, 544, 545, 548, 552, 553, 562, 564, 565, 566, 568, 574, 576, 580, 587, 588, 589, 590, 591, 592, 593, 594, 595, 597, 598, 599  
 ObliqueCartesianCS 120, 134, 333, 343  
 obliqueCartesianCSRef 134  
 ObliqueCartesianCSRefType 120, 134, 333, 343  
 ObliqueCartesianCSType 134, 343  
 observable 7, 295  
 observation xxi, 7, 14, 99, 168, 213, 238, 239, 240, 241, 278, 293, 294, 296, 438, 449, 450, 480, 560, 601  
 Observation 239, 240, 241, 283, 294, 450  
 observation.xsd 14, 278, 294, 438, 449, 601  
 ObservationType 239, 241, 294, 450  
 OffsetCurve 69, 416, 507  
 OffsetCurveType 69, 416  
 OGC i, ii, xx, xxi, 1, 2, 9, 11, 13, 31, 35, 39, 98, 103, 109, 111, 124, 136, 150, 168, 171, 242, 249, 280, 285, 301, 307, 309, 325, 332, 335, 343, 351, 354, 356, 363, 374, 377, 380, 390, 399, 404, 407, 438, 444, 445, 451, 455, 461, 466, 478, 481, 487, 531, 569  
 operation 101, 102, 103, 110, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 191, 291, 298, 299, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 491, 493, 496, 514, 523, 532, 537, 540, 569  
 OperationMethod 158, 162, 164, 314, 319, 320  
 OperationMethodBaseType 162, 319  
 operationMethodRef 164  
 OperationMethodRefType 158, 164, 314, 320  
 OperationMethodType 162, 319  
 OperationParameter 161, 162, 165, 166, 167, 168, 318, 319, 321, 322, 323, 324  
 OperationParameterBaseType 165, 322  
 OperationParameterGroup 162, 166, 168, 319, 323, 324  
 OperationParameterGroupBaseType 166, 323  
 operationParameterGroupRef 167  
 OperationParameterGroupRefType 162, 319  
 OperationParameterGroupType 166, 323  
 operationParameterRef 166  
 OperationParameterRefType 161, 166, 167, 318, 322, 324  
 OperationParameterType 165, 321  
 operationRef 153, 154, 155, 313, 315  
 OperationRefType 153, 154, 155, 157, 311, 312, 313, 315  
 operationVersion 150, 151, 154, 309, 315  
 OrientableCurve 36, 68, 69, 407, 493, 498, 588  
 OrientableCurveType 68, 407, 588  
 OrientableSurface 36, 76, 77, 426, 493, 498, 588  
 OrientableSurfaceType 76, 77, 426, 588  
 origin 146, 197, 246, 247, 265, 358  
 outerBoundaryIs 52

**P**

parameter 5, 56, 66, 69, 70, 71, 78, 79, 123, 135, 137, 141, 147, 149, 150, 153, 154, 155, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 296, 308, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 352, 355, 362, 408, 409, 417, 418, 419, 422, 429, 430, 532, 600, 606  
 parameterID 165, 166, 322  
 parameterName 165, 322  
 parameterValue 159, 161  
 parameterValueGroup 161  
 ParameterValueGroupType 161, 318  
 ParameterValueType 158, 159, 314, 316  
 ParametricCurveSurface 77, 78  
 PassThroughOperation 156, 157, 311, 312  
 passThroughOperationRef 157  
 PassThroughOperationRefType 157, 312

PassThroughOperationType 156, 311  
 patches 73, 74, 75, 77, 81, 82, 83, 425, 432, 433, 496, 500  
 pattern 18, 23, 24, 25, 26, 30, 35, 39, 40, 45, 104, 173, 174, 175, 176, 177, 180, 181, 191, 200, 209, 211, 224, 232, 233, 234, 236, 240, 285, 286, 287, 288, 289, 290, 303, 378, 381, 382, 383, 384, 385, 389, 390, 393, 394, 396, 397, 400, 405, 407, 425, 426, 435, 436, 437, 438, 441, 442, 453, 460, 463, 470, 471, 472, 473, 476, 477, 478, 483, 484, 511, 525, 526, 569, 573, 574, 575, 576, 587, 589, 590, 591, 594, 598  
 pixelInCell 144, 356  
 PixelInCellType 144, 356  
 point 4, 5, 7, 11, 14, 26, 44, 45, 46, 48, 49, 50, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 68, 69, 71, 72, 80, 81, 87, 88, 92, 98, 99, 109, 121, 122, 123, 130, 131, 133, 135, 138, 143, 148, 151, 168, 172, 173, 177, 185, 190, 193, 204, 218, 237, 242, 243, 250, 255, 257, 263, 264, 309, 340, 342, 347, 348, 355, 356, 382, 391, 393, 395, 402, 404, 407, 408, 409, 410, 411, 412, 413, 414, 415, 417, 419, 420, 421, 423, 424, 430, 432, 459, 474, 484, 485, 511, 531  
 Point 25, 26, 34, 36, 37, 48, 49, 77, 79, 87, 92, 206, 233, 234, 246, 247, 258, 259, 265, 288, 379, 388, 393, 394, 430, 445, 471, 497, 502, 514, 579, 588, 595  
 pointArrayProperty 36, 49  
 PointArrayPropertyType 36, 49, 92, 388, 394  
 PointGrid 77, 79, 430, 509  
 pointMember 91, 92, 258, 259, 382  
 pointMembers 91, 92, 382  
 pointProperty 36, 45, 48, 49, 50, 53, 54, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 72, 172, 393, 396, 397, 402, 410, 411, 412, 413, 414, 415, 421, 423, 424, 471, 502, 517  
 Point.PropertyType 36, 37, 48, 49, 50, 54, 57, 58, 59, 60, 61, 63, 64, 66, 68, 72, 92, 246, 379, 388, 393, 445, 471, 579  
 pointRep 57, 58, 60, 61, 62, 64, 65, 67, 396, 402, 410, 411, 412, 413, 414, 415, 421, 423, 424  
 PointType 48, 288, 393, 579, 588, 595  
 PolarCS 132, 133, 342  
 polarCSRef 133  
 PolarCSRefType 133, 342  
 PolarCSType 132, 342  
 polygon 7, 11, 52, 54, 74, 75, 81, 82, 96, 106, 242, 274, 389, 401, 403, 427, 428, 432, 433, 455  
 Polygon 36, 37, 52, 54, 74, 82, 96, 106, 261, 262, 389, 401, 403, 427, 432, 433, 455, 493, 509, 553, 588  
 polygonMember 95, 387  
 PolygonPatch 52, 74, 82, 427, 432, 433, 509  
 PolygonPatchArrayType 82, 432  
 polygonPatches 81, 82, 432  
 PolygonPatchType 74, 427  
 polygonProperty 54, 600, 606  
 Polygon.PropertyType 54, 96, 389, 403  
 PolygonType 52, 106, 401, 455, 588  
 PolyhedralSurface 81, 432, 510  
 PolyhedralSurfaceType 81, 432  
 pos 25, 26, 34, 44, 45, 46, 47, 48, 50, 51, 53, 54, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 72, 83, 205, 206, 233, 234, 247, 258, 259, 260, 261, 262, 265, 393, 396, 397, 398, 402, 403, 410, 411, 412, 413, 414, 415, 420, 421, 423, 424, 435, 497, 502  
 position 4, 5, 7, 25, 26, 32, 37, 38, 43, 44, 45, 47, 48, 57, 58, 61, 63, 65, 66, 68, 70, 72, 75, 98, 99, 100, 101, 109, 113, 122, 123, 128, 129, 130, 133, 138, 145, 151, 157, 168, 169, 182, 183, 184, 185, 186, 187, 188, 189, 191, 192, 193, 194, 196, 197, 198, 199, 200, 202, 204, 205, 231, 242, 246, 248, 255, 497, 521, 523, 525, 526, 528  
 posList 44, 45, 48, 50, 53, 54, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 72, 83, 260, 261, 262, 396, 397, 398, 403, 410, 411, 412, 413, 414, 415, 420, 421, 423, 424, 435  
 PrimeMeridian 139, 142, 143, 359, 360, 361  
 PrimeMeridianBaseType 142, 360  
 primeMeridianRef 143  
 PrimeMeridianRefType 139, 143, 359, 361  
 PrimeMeridianType 142, 360  
*priorityLocation* 34, 35  
 PriorityLocation.PropertyType 34, 380  
 profile xx, 22, 182, 280, 281, 282, 284, 490, 491, 492, 493, 495, 501, 511, 512, 513, 514, 516, 520, 521, 522, 523, 526, 527, 529, 541, 544, 545, 547, 565, 571, 574, 575, 593, 597  
 Projected 101, 109, 110, 122, 124, 531  
 ProjectedCRS 116, 117, 330, 331  
 projectedCRSRef 117  
 ProjectedCRSRefType 117, 330

ProjectedCRSType 116, 330

property xxi, 1, 3, 4, 6, 7, 8, 11, 12, 13, 14, 15, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 43, 44, 47, 48, 49, 50, 51, 52, 53, 54, 56, 66, 68, 70, 74, 76, 77, 82, 83, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 107, 108, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 126, 128, 129, 130, 131, 132, 133, 134, 139, 141, 143, 144, 145, 146, 152, 153, 154, 155, 156, 157, 158, 159, 164, 165, 166, 168, 171, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 186, 187, 189, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 232, 234, 236, 239, 240, 241, 243, 244, 246, 247, 248, 249, 250, 251, 252, 254, 256, 263, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 280, 283, 284, 285, 286, 287, 288, 289, 290, 291, 294, 296, 301, 345, 363, 364, 369, 370, 372, 375, 377, 378, 379, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 409, 418, 422, 423, 424, 425, 426, 428, 429, 432, 433, 435, 436, 439, 441, 442, 443, 444, 449, 461, 463, 467, 468, 469, 478, 485, 486, 491, 493, 495, 496, 497, 498, 499, 500, 501, 502, 503, 505, 506, 507, 509, 511, 512, 513, 514, 516, 517, 519, 520, 522, 523, 524, 525, 526, 527, 528, 529, 530, 532, 533, 535, 536, 539, 540, 541, 542, 543, 544, 552, 553, 555, 560, 564, 565, 569, 570, 573, 574, 575, 576, 577, 578, 579, 582, 583, 587, 589, 590, 591, 593, 594, 597, 598, 599

## Q

QName 19

QNameList 19

quantity xxi, 7, 212, 213, 214, 215, 216, 219, 220, 224, 225, 296, 446, 447, 479, 480, 492

Quantity 231, 233, 234, 235, 236, 240, 250, 292, 293, 484, 486

QuantityExtent 235, 484

QuantityExtentType 235, 484

QuantityList 231, 233, 235, 250, 292, 293

Quantity.PropertyType 236

quantityType 213, 217, 218, 219, 220, 221, 222, 223, 479

QueryGrammarEnumeration 270, 365

## R

range xxi, 5, 6, 7, 30, 104, 210, 212, 238, 242, 243, 244, 248, 250, 251, 252, 254, 258, 292, 344, 345, 347, 453, 484, 485, 545

rangeParameters 250, 251, 252, 253, 256, 260, 262, 264, 265, 346

RangeParametersType 250, 346

rangeSet 247, 250, 256, 258, 259, 260, 261, 262, 263, 264, 265, 344, 348, 349, 350, 351

RangeSetType 250, 345

RDF 9, 21

realization 6, 138, 355, 491, 514, 521

realizationEpoch 137, 138, 355

Rectangle 75, 428, 554

RectangleType 75, 428

rectified grid 7, 246, 264, 445

RectifiedGrid 244, 246, 247, 264, 265, 351, 445

RectifiedGridCoverage 264, 351

RectifiedGridCoverageType 351

rectifiedGridDomain 264, 265, 351

RectifiedGridDomainType 351

RectifiedGridType 246, 445

reference 26, 192, 200, 210, 214, 578, 589, 590

reference system xxi, xxii, 4, 5, 6, 7, 12, 19, 33, 34, 41, 42, 43, 44, 45, 47, 50, 54, 57, 58, 59, 60, 61, 64, 66, 68, 70, 71, 72, 78, 97, 98, 99, 100, 101,

102, 103, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 125, 126, 127, 134, 135, 136, 137, 140,

142, 143, 144, 146, 147, 148, 150, 151, 153, 155, 156, 158, 163, 165, 167, 168, 182, 185, 188, 195, 197, 199, 201, 207, 249, 277, 282, 291, 292,

294, 295, 310, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 354, 355, 391, 417, 418, 419, 420, 451, 452, 453, 454, 457, 460, 462, 463, 527,

529, 531, 545, 564

referenceSystem 107, 108, 234, 453, 484

referenceSystemRef 107, 108, 453

ReferenceSystemRefType 107, 108, 452, 453

referenceSystems.xsd 97, 103, 124, 137, 307, 308, 325, 335, 354, 450, 451, 601

ReferenceType 26, 192, 200, 210, 214, 373, 442, 463, 467, 479, 578, 589, 590

RelatedTimeType 183, 184, 200, 456, 463

relativeInternalPositionalAccuracy 169

RelativeInternalPositionalAccuracyType 169, 352

remarks 104, 106, 125, 127, 137, 140, 142, 150, 153, 154, 162, 165, 166, 308, 313, 315, 319, 322, 323, 335, 337, 354, 360, 361, 451, 453

remoteSchema 25, 27, 268, 442, 443

result 169, 170, 239, 240, 241, 242, 296, 352, 450  
 resultOf 239, 240, 241, 242, 296, 450  
 RFC 2, 9  
 ring 53, 54, 75, 76, 174, 401, 402, 403, 427, 428, 429, 553  
 Ring 4, 5, 6, 7, 52, 53, 56, 73, 74, 75, 76, 83, 86, 99, 176, 261, 262, 428, 429, 496, 501, 509, 514, 553, 607  
 RingPropertyType 76  
 RingType 75, 76, 428  
 role 25, 488  
 roughConversionToPreferredUnit 215, 216, 479  
 rowIndex 170, 171, 353

## S

ScaleType 141, 224, 363, 365, 492, 572, 597  
 schema 1, 3, 8, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 35, 37, 38, 39, 40, 41, 46, 55, 56, 74, 86, 97, 98, 103, 104, 108, 110, 111, 121, 124, 134, 136, 137, 147, 149, 150, 168, 169, 171, 182, 183, 185, 190, 191, 192, 195, 197, 201, 202, 203, 204, 205, 206, 207, 208, 211, 212, 213, 216, 224, 225, 227, 228, 229, 230, 231, 232, 233, 236, 237, 238, 239, 240, 244, 245, 246, 247, 248, 249, 251, 266, 267, 268, 269, 270, 271, 272, 275, 277, 278, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 302, 303, 307, 324, 325, 334, 335, 343, 351, 353, 354, 363, 371, 373, 374, 376, 377, 380, 389, 399, 404, 406, 407, 438, 442, 443, 444, 445, 446, 449, 450, 451, 455, 457, 461, 463, 466, 469, 470, 478, 481, 482, 484, 487, 489, 490, 498, 529, 530, 532, 533, 535, 541, 543, 547, 548, 560, 562, 568, 569, 570, 571, 581, 583, 585, 586, 587, 589, 590, 591, 592, 593, 594, 595, 598, 599, 600, 601, 602, 607, 608  
 Schematron 3, 26, 234, 286, 389, 438, 442, 484, 568, 591, 594  
 scope 105, 106, 137, 146, 150, 153, 154, 309, 313, 315, 355, 358, 452  
 secondDefiningParameter 140, 141, 362  
 SecondDefiningParameterType 141, 362  
 seconds 189, 225, 226, 227, 448  
 segments 6, 55, 56, 64, 68, 72, 84, 407, 499, 504, 552, 554  
 semantic 3, 8, 39, 40, 210, 250, 283, 285, 287, 288, 294, 469, 487, 507  
 semantic type 8, 39, 40, 287, 294  
 semiMajorAxis 140, 141, 362  
 semiMinorAxis 141, 362  
 sequence 4, 8, 11, 24, 26, 28, 29, 31, 32, 33, 34, 35, 39, 40, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 79, 81, 82, 83, 84, 85, 86, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 102, 104, 105, 106, 107, 108, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 123, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 148, 149, 150, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 164, 165, 166, 167, 168, 169, 170, 172, 173, 174, 175, 177, 178, 179, 180, 181, 182, 183, 184, 186, 187, 192, 193, 194, 196, 198, 199, 200, 201, 203, 204, 205, 206, 207, 208, 209, 212, 213, 214, 215, 216, 225, 232, 236, 237, 239, 241, 242, 245, 246, 247, 248, 249, 250, 251, 252, 255, 258, 259, 261, 262, 263, 264, 267, 268, 269, 270, 271, 272, 273, 276, 288, 289, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 381, 382, 383, 384, 385, 386, 387, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 432, 433, 434, 435, 436, 437, 439, 440, 441, 442, 444, 445, 447, 448, 450, 451, 452, 453, 454, 455, 456, 458, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 483, 485, 486, 487, 525, 545, 572, 573, 574, 575, 576, 577, 578, 582, 583, 589, 590, 593, 598  
 SequenceRuleNames 255, 347  
 SequenceRuleType 255, 347  
 serialization 298, 299  
 set ii, xxi, 3, 4, 5, 6, 7, 8, 10, 15, 16, 18, 19, 35, 36, 38, 39, 40, 41, 47, 64, 66, 74, 80, 81, 84, 87, 88, 98, 99, 100, 101, 107, 108, 110, 121, 123, 125, 126, 127, 135, 136, 137, 138, 140, 142, 145, 149, 151, 158, 159, 162, 163, 165, 167, 168, 170, 172, 174, 175, 178, 179, 184, 191, 193, 197, 200, 203, 206, 207, 210, 212, 215, 224, 234, 235, 236, 242, 243, 244, 245, 246, 248, 250, 251, 252, 257, 263, 265, 267, 269, 270, 272, 275, 276, 277, 280, 286, 290, 294, 295, 297, 307, 309, 314, 316, 318, 320, 324, 336, 337, 344, 345, 347, 348, 349, 353, 355, 357, 366, 371, 377, 381, 382, 383, 384, 385, 386, 387, 390, 392, 393, 394, 395, 396, 397, 398, 399, 400, 403, 404, 406, 421, 423, 424, 425, 427, 431, 434, 436, 440, 444, 471, 472, 473, 475, 486, 504, 505, 508, 538, 539, 542, 543, 544, 562, 563, 569, 570, 575, 576, 585, 586, 592, 595, 598, 600  
 show 25, 488  
 SignType 17, 68, 76, 173, 174, 175, 176, 408, 426, 448, 471, 472, 473, 474, 572, 575  
 simple feature 11  
 simpleLink 25, 443  
 SimpleNameType 104, 107, 125, 127, 138, 140, 142, 151, 163, 165, 167, 308, 319, 322, 323, 335, 337, 354, 360, 361, 451  
 singleOperationRef 152, 153, 157, 312

SingleOperationRefType 152, 153, 156, 157, 310, 311, 312  
SMIL 3, 9, 266, 275, 585  
smil20 363, 366, 542, 585, 600, 607, 608  
smil20.xsd 363, 607  
smil20Lang 585, 600, 607  
smil20-language.xsd 607  
SOAP 9, 252  
Solid 36, 85, 86, 89, 97, 388, 435, 436, 493, 499, 501, 519, 579, 580, 588, 595, 596  
solidArrayProperty 36, 85, 86  
SolidArrayType 36, 85, 97, 388, 436  
solidMember 89, 96, 97, 385, 405  
solidMembers 96, 97, 385  
solidProperty 36, 85, 175, 435, 517  
SolidPropertyType 36, 85, 97, 388, 435, 579  
SolidType 86, 436, 579, 588, 595  
sourceCRS 150, 151, 154, 309, 315  
sourceDimensions 163, 320  
spatial object 6, 8  
SpeedType 492, 572, 597  
Sphere 80, 431, 510  
SphereType 80, 431  
SphericalCS 116, 132, 329, 341, 342  
sphericalCSRef 132  
SphericalCSRefType 116, 132, 329, 341  
SphericalCSType 132, 341  
srsDimension 42, 43, 45, 47  
srsDimension 47  
srsID 106, 107, 452  
SRSInformationGroup 42, 43, 391  
srsName 25, 33, 34, 41, 42, 43, 44, 45, 105, 106, 107, 233, 234, 247, 258, 260, 261, 265, 451, 495, 502, 532, 545  
SRSReferenceGroup 42, 44, 45, 47, 245, 391, 396, 397, 398  
StandardObjectProperties 28, 39, 81, 82, 258, 259, 261, 262, 263, 264, 348, 349, 350, 351, 379, 432, 433, 439  
status 205, 375  
sterotype 17  
stringOrNull 18, 572  
stringOrRefType 27  
StringOrRefType 27, 28, 34, 104, 198, 199, 203, 205, 213, 236, 254, 347, 374, 375, 379, 443, 454, 464, 465, 480  
stringValue 160, 317  
Style 267, 268, 269, 364, 365, 538, 539, 541  
StyleType 268, 364  
StyleVariationType 365  
subComplex 179, 180, 476  
subject 240, 241, 242  
subtype 3, i, 42, 75, 79, 87, 107, 108, 111, 112, 127, 137, 325, 329, 337, 355, 427, 428, 430, 452, 459, 493, 531, 552, 553  
SuccessionType 195  
superComplex 179, 180, 476  
supertype 42, 192, 390, 391, 439, 440, 442, 456, 457, 470, 530, 565, 571, 572, 573, 574, 578  
Surface 36, 37, 51, 52, 73, 74, 75, 76, 77, 81, 82, 83, 86, 88, 95, 379, 388, 400, 425, 426, 427, 428, 432, 433, 436, 496, 498, 499, 500, 553, 574, 579, 582, 588, 595  
surfaceArrayProperty 36, 52  
SurfaceArrayPropertyType 36, 52, 95, 388, 400  
SurfaceInterpolationType 74, 75, 427, 428  
surfaceMember 88, 94, 95, 96, 261, 262, 384, 405, 607  
surfaceMembers 94, 95, 384  
SurfacePatch 73, 74, 75, 77, 588  
SurfacePatchArrayType 73, 82, 426, 433  
surfaceProperty 36, 51, 52, 54, 174, 400, 473, 517, 607  
SurfacePropertyType 36, 37, 51, 52, 54, 76, 86, 95, 379, 388, 400, 425, 436, 574, 579, 582  
SurfaceType 73, 81, 82, 425, 432, 433, 579, 588, 595  
SVG 3, 9, 274, 275, 533, 539, 540, 542, 543

symbol 271, 272, 274, 275, 366, 367, 540, 541, 542

SymbolType 369

SymbolTypeEnum 369

## T

tag 5, 8, 563

target 150, 151, 154, 163, 239, 240, 241, 309, 315, 320, 449, 450, 530, 578, 599

targetCRS 150, 151, 154, 309, 315

targetDimensions 163, 320

TargetPropertyType 239, 449

Temporal 2, 38, 100, 109, 111, 122, 136, 182, 183, 184, 185, 191, 192, 193, 195, 231, 284, 290, 325, 345, 457, 461, 466, 482, 521, 522, 523, 524, 525, 526, 527, 529, 531, 545

temporal reference system xxii, 12, 106, 182, 185, 188, 195, 197, 199, 200, 207, 452, 457, 461, 462, 463, 464, 527

temporal.xsd 104, 203, 228, 229, 290, 377, 451, 455, 466, 481, 601

TemporalCRS 120, 121, 333, 334

temporalCRSRef 121

TemporalCRSRefType 121, 334

TemporalCRSType 120, 333

TemporalCS 120, 130, 334, 340

temporalCSRef 130

TemporalCSRefType 120, 130, 334, 340

TemporalCSType 130, 340

TemporalDatum 121, 146, 334, 358, 359

TemporalDatumBaseType 146, 358

temporalDatumRef 146

TemporalDatumRefType 121, 146, 334, 359

TemporalDatumType 146, 358

temporalExtent 105, 106, 454

temporalReferenceSystems.xsd 278, 438, 461

temporalTopology.xsd 461, 466

test 26, 181, 233, 234, 282, 297, 298, 299, 300, 301, 389, 390, 438, 476, 484, 542, 543, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612

TimeCalendar 38, 195, 197, 198, 199, 464, 465, 466, 528

TimeCalendarEra 38, 195, 197, 198, 464, 465, 528

TimeCalendarEra.PropertyType 38, 198, 464

TimeCalendarEraType 198, 464

TimeCalendar.PropertyType 38, 199, 466

TimeCalendarType 197, 198, 464

TimeClock 39, 195, 199, 465, 466, 528

TimeClock.PropertyType 39

TimeClockType 199, 465

TimeComplex 185, 193

TimeCoordinateSystem 195, 196, 197, 462, 528

TimeCoordinateSystemType 196, 462

TimeEdge 38, 193, 194, 200, 468, 469, 527, 529, 580, 589, 596

TimeEdge.PropertyType 38, 193, 194, 468, 580

TimeEdgeType 194, 200, 468, 580, 589, 596

TimeGeometricPrimitive 185, 186

TimeGeometricPrimitive.PropertyType 38

TimeIndeterminateValueType 188, 460

TimeInstant 38, 185, 186, 187, 189, 194, 196, 197, 201, 206, 233, 240, 241, 242, 290, 458, 462, 468, 523, 528, 580, 588, 596

TimeInstant.PropertyType 38, 186, 187, 194, 196, 197, 290, 458, 462, 468, 580

TimeInstantType 186, 458, 580, 588, 596

timeInterval 38, 190, 191, 524

TimeIntervalLengthType 38, 190, 196, 197, 459, 462, 528

TimeLengthType 190, 459

TimeNode 38, 193, 194, 200, 463, 468, 469, 526, 580, 589, 596

TimeNode.PropertyType 38, 194, 200, 463, 469, 580

TimeNode.Type 193, 468, 580, 589, 596

TimeObject 183, 185, 290  
TimeOrdinalEra 38, 195, 200, 201, 202, 463, 464, 529  
TimeOrdinalEraPropertyType 38, 200, 463  
TimeOrdinalEraType 200, 463  
TimeOrdinalReferenceSystem 195, 200, 201, 202, 462, 529  
TimeOrdinalReferenceSystemType 200, 462  
TimePeriod 38, 106, 186, 187, 194, 198, 200, 455, 458, 463, 465, 469, 523, 580, 589, 596  
TimePeriodPropertyType 38, 187, 194, 198, 200, 463, 465, 469, 580  
TimePeriodType 106, 186, 455, 458, 580, 589, 596  
timePosition 33, 38, 186, 187, 188, 189, 197, 201, 206, 233, 240, 241, 242, 378, 458, 523  
TimePositionType 186, 187, 188, 189, 196, 458, 460, 462, 525  
TimePositionUnion 188, 189, 460  
TimePrimitive 183, 185, 192  
TimePrimitivePropertyType 38, 184, 186, 187, 456, 461  
timeslice 204, 375  
TimeTopologyComplex 38, 193, 466, 467, 523, 526, 580, 589, 596  
TimeTopologyComplexPropertyType 38, 192, 580  
TimeTopologyComplexType 193, 466, 580, 589, 596  
TimeTopologyPrimitive 192  
TimeTopologyPrimitivePropertyType 38, 192, 193, 467  
TimeType 224, 492  
TimeUnitType 190, 459  
Tin 83, 434, 510  
TinType 83, 434  
title 25, 488, 489  
to 488  
TopoComplex 179, 180, 182, 289, 476, 477, 580, 596  
TopoComplexMemberType 180, 182, 289, 290, 476, 580  
topoComplexProperty 182  
TopoComplexType 179, 180, 476, 580, 596  
TopoCurve 178, 244, 474  
topoCurveProperty 178  
TopoCurve.PropertyType 178, 474  
TopoCurveType 178, 474  
topological object 8, 18, 304  
topology xxi, 8, 12, 14, 37, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 184, 191, 192, 193, 244, 269, 270, 271, 272, 274, 275, 277, 278, 289, 290, 294, 365, 367, 368, 375, 438, 457, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 514, 519, 520, 526, 541, 555, 601  
topology.xsd 14, 278, 289, 290, 438, 470, 601  
topologyStyle 270, 271, 365, 542  
TopologyStyle 270, 271, 272, 367, 541, 542  
TopologyStyle.PropertyType 271, 367  
TopologyStyleType 272, 367  
TopoPoint 177, 244, 474  
topoPointProperty 177  
TopoPoint.PropertyType 177, 474  
TopoPointType 177, 474  
TopoPrimitiveArrayAssociationType 181, 477  
topoPrimitiveMember 180, 181, 476, 477  
topoPrimitiveMembers 180, 181, 476  
topoPrimitiveMemberType 181, 477  
TopoSolid 175, 177, 244, 471, 473, 474  
TopoSolidType 175, 473  
TopoSurface 178, 179, 244, 475  
topoSurfaceProperty 178  
TopoSurface.PropertyType 178, 179, 475  
TopoSurfaceType 178, 475  
TopoVolume 179, 289, 475  
topoVolumeProperty 179  
TopoVolume.PropertyType 179, 289, 475  
TopoVolumeType 179, 475

track 206, 606  
TrackType 206, 376  
transform 159, 273, 369, 370  
Transformation 158, 159, 315, 316  
transformationRef 159  
TransformationRefType 159, 316  
TransformationType 158, 315  
Triangle 74, 75, 82, 427, 433  
TrianglePatchArrayPropertyType 82, 433  
trianglePatches 82, 433  
TriangleType 74, 75, 427  
TriangulatedSurface 82, 83, 433, 434, 510  
TriangulatedSurfaceType 82, 83, 433, 434  
tuple 4, 8, 46, 48, 98, 99, 109, 110, 115, 122, 123, 126, 127, 149, 156, 157, 251, 307, 345  
tupleList 250, 251, 256, 346  
type 3, i, xxi, 5, 6, 7, 8, 10, 11, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 79, 80, 81, 82, 83, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 99, 100, 101, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 196, 197, 198, 199, 200, 203, 204, 205, 206, 207, 208, 209, 210, 212, 213, 214, 215, 216, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 239, 240, 241, 242, 244, 245, 246, 247, 248, 249, 250, 251, 252, 254, 255, 257, 263, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 293, 294, 295, 301, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 492, 493, 497, 498, 502, 511, 514, 521, 524, 525, 526, 527, 530, 536, 537, 539, 560, 562, 564, 565, 566, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 582, 583, 586, 587, 589, 590, 591, 593, 594, 595, 597, 598, 599, 602, 603, 604, 605, 606

## U

UML 9, 10, 12, 17, 22, 25, 26, 98, 101, 103, 111, 124, 136, 150, 168, 202, 210, 211, 228, 266, 307, 309, 325, 335, 351, 354, 442, 451, 490, 491, 492, 493, 495, 501, 511, 512, 514, 516, 520, 521, 522, 525, 527, 529, 530, 531, 534, 535, 536, 537, 538, 539, 541, 543, 544, 545, 546, 547, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 578, 579, 585, 586, 592, 593, 594, 595, 597, 598, 599  
UnitDefinition 213, 214, 215, 224, 478, 479, 492  
UnitDefinitionType 213, 214, 215, 478, 479, 492  
unitDerivation 214, 215  
UnitDerivationType 215  
unitOfMeasure 170, 212, 213, 353  
UnitOfMeasureType 212, 215, 216, 478, 480, 572  
units 19, 21, 43, 125, 126, 206, 211, 212, 213, 215, 216, 218, 219, 220, 221, 222, 223, 224, 225, 227, 231, 233, 234, 235, 237, 240, 242, 251, 252, 253, 256, 259, 260, 262, 264, 265, 275, 492  
units.xsd 14, 169, 212, 351, 352, 390, 445, 446, 478, 601  
unitTerm 215, 218, 219, 220, 221, 222, 223  
 uom 19, 21, 43, 125, 126, 206, 211, 212, 213, 215, 216, 218, 219, 220, 221, 222, 223, 224, 225, 227, 231, 233, 234, 235, 237, 240, 242, 251, 252, 253, 256, 259, 260, 262, 264, 265, 275, 336, 492  
URI 2, 7, 8, 15, 16, 17, 29, 30, 42, 149, 185, 188, 189, 207, 211, 213, 224, 230, 231, 268, 285, 303, 391, 443, 457, 460, 482, 483, 492, 541, 563, 570, 578, 586  
URL 9, 284, 532, 543  
UserDefinedCS 131, 132, 341  
userDefinedCSRef 131  
UserDefinedCSRefType 131, 341  
UserDefinedCSType 131, 341  
usesAxis 127, 128, 337  
usesCartesianCS 116, 117, 119, 328, 330, 333  
usesCS 117, 118, 331, 332

usesEllipsoid 114, 139, 327, 359  
usesEllipsoidalCS 114, 327  
usesEngineeringDatum 118, 119, 332  
usesGeodeticDatum 114, 116, 327, 328  
usesImageDatum 119, 120, 333  
usesMethod 157, 158, 314, 316  
usesObliqueCartesianCS 119, 120, 333  
usesOperation 156, 157, 311  
usesParameter 162, 163, 320  
usesPrimeMeridian 139, 359  
usesSingleOperation 155, 156, 310  
usesSphericalCS 116, 328  
usesTemporalCS 120, 334  
usesTemporalDatum 120, 121, 334  
usesValue 157, 158, 314, 316  
usesVerticalCS 115, 328  
usesVerticalDatum 115, 328  
using 239, 240, 241, 242, 450  
UTC xxii, 185, 188, 195, 199, 457, 466

## V

validArea 105, 106, 137, 146, 150, 153, 154, 309, 313, 315, 355, 358, 452  
validate 103, 105  
validTime 38, 184, 203, 204, 206, 239, 240, 241, 242, 375, 376, 450  
value 159, 160, 161, 162, 232, 233, 234, 251, 252, 253, 256, 259, 260, 262, 264, 265, 317, 318, 483, 484  
Value 231, 232, 233, 234, 235, 236, 250, 259, 292, 345, 484, 485, 486, 487  
ValueArray 232, 233, 234, 235, 250, 259, 345, 484, 486  
ValueArrayType 232, 486  
ValueArrayType 233, 484  
valueComponent 232, 233, 234, 251, 252, 253, 256, 259, 260, 262, 264, 265, 483, 484  
valueComponents 232, 233, 234, 251, 252, 253, 256, 259, 260, 264, 265, 483, 484  
valueFile 160, 161, 317  
valueList 160, 161, 317  
valueObjects.xsd 14, 212, 228, 229, 292, 296, 344, 449, 481, 601  
valueOfParameter 160, 161, 317  
valueProperty 232  
Value.PropertyType 232, 236, 485, 486, 487  
valuesOfGroup 161, 162, 318  
vector 47, 60, 62, 65, 69, 70, 71, 78, 99, 100, 237, 243, 374, 397, 492, 533  
VectorType 46, 47, 60, 61, 64, 69, 70, 246, 397, 413, 414, 417, 418, 421, 445, 492, 571, 575, 583, 597  
verification 12  
version 104, 453  
Vertical 109, 110, 115, 122, 124, 135, 145, 327, 357, 358  
VerticalCRS 114, 115, 327, 328  
verticalCRSRef 115  
VerticalCRSRefType 115, 328  
VerticalCRSType 114, 327  
VerticalCS 115, 129, 130, 328, 339, 340  
verticalCSRef 130  
VerticalCSRefType 115, 130, 328, 339  
VerticalCSType 129, 339  
VerticalDatum 115, 145, 328, 357, 358  
verticalDatumRef 145  
VerticalDatumRefType 115, 145, 328, 358  
verticalDatumType 145, 357  
VerticalDatumType 145, 357  
VerticalDatumTypeType 145, 357  
verticalExtent 105, 106, 454  
vocabulary 3, 101, 109, 283, 284, 286, 287, 289, 291, 292, 293, 295, 296

VolumeType 224, 492, 572, 597

## W

WFS 10  
WKT 9

## X

xlink 7, 15, 24, 25, 26, 27, 34, 40, 187, 196, 197, 201, 210, 211, 217, 218, 234, 238, 240, 241, 242, 249, 254, 275, 277, 286, 303, 343, 373, 380, 389, 399, 404, 407, 438, 439, 443, 444, 466, 470, 481, 487, 488, 489, 536, 571, 581, 585, 600, 601, 606, 607, 608  
Xlinks 6, 7, 13, 15, 21, 22, 24, 25, 26, 27, 34, 37, 99, 100, 101, 110, 135, 173, 174, 175, 176, 187, 196, 197, 201, 210, 211, 217, 218, 234, 238, 240, 241, 242, 249, 254, 275, 282, 288, 495, 497, 498, 499, 500, 502, 511, 512, 513, 514, 517, 519, 520, 526, 527, 528, 529, 535, 536, 540, 544, 570, 572, 573, 576, 599, 601, 606  
xlinks.xsd 15, 25, 268, 277, 381, 382, 383, 384, 385, 386, 387, 390, 392, 394, 395, 399, 400, 403, 406, 436, 439, 443, 487, 581, 607  
xml xxi, 1, 2, 3, 5, 7, 8, 10, 11, 12, 13, 15, 17, 18, 20, 21, 22, 23, 24, 25, 26, 27, 28, 30, 31, 33, 36, 37, 38, 39, 41, 46, 54, 55, 86, 87, 90, 101, 102, 103, 110, 124, 135, 136, 149, 161, 168, 184, 187, 188, 189, 190, 207, 208, 209, 210, 213, 216, 217, 224, 227, 228, 229, 230, 231, 252, 275, 278, 280, 283, 284, 285, 286, 287, 290, 291, 292, 294, 295, 297, 298, 299, 301, 302, 303, 304, 305, 307, 318, 324, 334, 343, 351, 354, 363, 371, 372, 373, 374, 375, 376, 377, 380, 381, 382, 383, 384, 385, 386, 387, 389, 390, 392, 394, 395, 399, 400, 403, 404, 406, 407, 436, 438, 439, 441, 442, 443, 444, 445, 446, 448, 449, 450, 451, 455, 456, 457, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 478, 481, 482, 483, 487, 490, 491, 498, 512, 522, 524, 525, 526, 527, 529, 530, 532, 533, 536, 547, 548, 552, 553, 560, 562, 563, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 578, 581, 585, 586, 587, 589, 591, 592, 593, 594, 597, 599, 600, 601, 606, 607, 609, 614  
content model 16, 17, 24, 25, 27, 28, 30, 32, 33, 34, 35, 39, 40, 41, 45, 183, 184, 185, 186, 188, 190, 192, 193, 194, 195, 196, 197, 198, 199, 200, 203, 208, 209, 213, 215, 216, 224, 225, 227, 228, 229, 234, 235, 237, 239, 247, 248, 249, 250, 254, 255, 257, 259, 260, 262, 263, 264, 268, 269, 273, 276, 277, 283, 284, 285, 286, 292, 294, 295, 301, 439, 440, 442, 443, 484, 576, 587, 590, 593, 594  
instance document 12, 16, 17, 18, 19, 20, 21, 27, 28, 186, 196, 207, 211, 224, 225, 226, 227, 231, 282, 283, 284, 285, 286, 287, 289, 290, 291, 292, 293, 294, 295, 296, 363, 439, 480, 491, 547, 563, 569, 570, 571, 573, 581, 585, 586, 587, 589, 591, 592, 593, 594, 595, 597, 600, 601, 606, 607, 609, 614  
Namespace 7, 9, 15, 98, 214, 267, 277, 278, 281, 282, 283, 284, 285, 286, 287, 289, 290, 291, 292, 293, 294, 295, 296, 363, 439, 480, 491, 547, 563, 569, 570, 571, 573, 581, 585, 586, 587, 589, 591, 592, 593, 594, 595, 597, 600, 601, 606, 607, 609, 614  
schemaLocation 171, 256, 258, 260, 261, 265, 285, 286, 287, 289, 290, 291, 292, 294, 295, 296, 308, 325, 335, 343, 344, 352, 354, 363, 371, 373, 375, 377, 380, 390, 399, 404, 407, 438, 439, 444, 446, 449, 451, 455, 461, 466, 470, 478, 481, 571, 581, 586, 607, 612  
substitution group 23, 24, 25, 27, 31, 32, 33, 39, 40, 42, 43, 48, 49, 51, 90, 101, 183, 184, 185, 190, 192, 206, 225, 228, 231, 232, 236, 240, 248, 249, 250, 281, 284, 285, 292, 295, 344, 364, 390, 391, 406, 409, 410, 411, 412, 414, 415, 416, 418, 420, 422, 424, 439, 441, 455, 456, 457, 459, 461, 467, 482, 499, 536, 591  
XML Linking Language 2, 13, 21, 25, 43, 48, 49, 51, 52, 54, 68, 76, 77, 85, 90, 91, 92, 93, 94, 95, 96, 97, 268, 280, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 392, 393, 394, 395, 399, 400, 403, 406, 407, 426, 428, 435, 436, 467, 468, 469, 488, 533, 569, 579  
XML Pointer 2, 31, 209, 373, 443, 446  
XML Schema xxi, 1, 2, 3, 11, 13, 15, 17, 18, 20, 22, 23, 24, 25, 26, 27, 30, 36, 37, 38, 39, 41, 46, 54, 55, 86, 87, 90, 101, 135, 187, 188, 189, 190, 216, 224, 228, 229, 231, 280, 283, 284, 285, 286, 287, 291, 292, 294, 295, 299, 302, 304, 305, 406, 442, 443, 446, 459, 460, 481, 482, 483, 490, 491, 498, 512, 522, 524, 525, 526, 527, 529, 530, 532, 533, 547, 548, 560, 562, 563, 567, 568, 569, 570, 571, 572, 575, 576, 578, 585, 586, 591, 592, 594, 597, 599  
XSLT 10, 539, 540, 600