TSMAD27-4.3.1F

S-100 – Part 7

Spatial Schema

Contents

7-1 Scope

The spatial requirements of S-100 are less comprehensive than the requirements of ISO 19107 “Geographical Information - Spatial schema” which contains all the information necessary for describing and manipulating the spatial characteristics of geographical features and on which this Part is based. Hence this Part contains only the subset of ISO 19107 classes required for S-100. This version only contains geometry, if there is a future requirement for topology then this Part will be extended to meet these requirements.
This Part specifies:

1) a subset of ISO 19107 classes (clause 6) which is the minimum required to support a 0, 1, 2 and 2.5 dimensional spatial schemas. As such it is restricted to specifying only data and does not include operations.
2) additional constraints (omitted optional elements or constrained cardinalities) which are imposed on these classes by this profile.

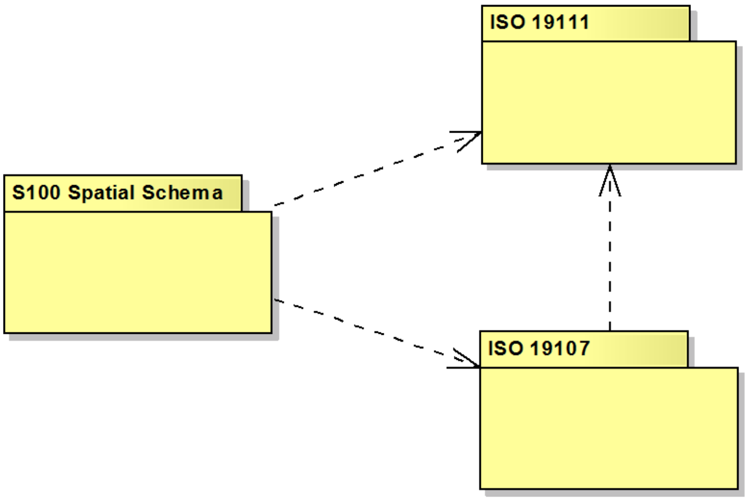


Figure 7-1 — S-100 Spatial Schema relationship with ISO 19100 Packages

7-2 Conformance

This profile consists of simple geometry based on three criteria – complexity, dimensionality and functional complexity. The first two criteria (complexity and dimensionality) determine the types defined in this profile that shall be implemented according to an application schema that conforms to a given conformance option.

There are:

two levels of complexity:

1) Geometric Primitives
2) Geometric Complexes,

four levels of dimensionality:

1) 0-dimensional objects
2) 0- and 1-dimensional objects
3) 0-, 1- and 2-dimensional objects
4) 0-, 1-, 2- and 2½ -dimensional objects

and one level of functional complexity:

1) data types only (operations are not included).

This profile satisfies the conformance classes A.1.1.1, A.1.1.2, A.1.1.3, A.2.1.1 and A.2.1.2 in ISO 19107. This profile conforms to level 2 of ISO 19106:2004.

7-3 Normative references

The following normative documents contain provisions that, through reference in this text, constitute provisions of this profile.

ISO 19107 Geographic information — Spatial schema

ISO TS 19103 Geographic information — Conceptual schema language

ISO 19111 Geographic information — Spatial referencing by coordinates

7-4 Symbols, notation and abbreviated terms

7-4.1 Abbreviations

OCL	Object Constraint Language
2-D	Two-dimensional
2.5D	Two and a half dimensional

7-5 Geometry

7-5.1 Introduction

This profile consists of simple geometry which can be expressed in multiple configurations as described at clause 6.1.3.

7-5.1.1 S-100 spatial schema Geometry classes and their ISO 19107 reference

Coordinate Geometry	Geometry Primitive	Geometry Complex	Geometry Aggregate
DirectPosition (6.4.1)	GM_Curve (6.3.16)	GM_Complex (6.6.2)	GM_Aggregate (6.5.2)
CurveInterpolation (6.4.8)	GM_CurveBoundary (6.3.5)	GM_Composite (6.6.3)	GM_MultiPoint (6.5.4)
GM_CurveSegment (6.4.9)	GM_OrientableCurve (6.3.14)	GM_CompositeCurve (6.6.5)	
GM_Position (6.4.5)	GM_OrientableSurface (6.3.15)		
GM_Polygon (6.4.36)	GM_Point (6.3.11)		
GM_SurfacePatch (6.4.34)	GM_Primitive (6.3.10)		
SurfaceInterpolation (6.4.32)	GM_Ring (6.3.6)		
	GM_Surface (6.3.17)		
	GM_SurfaceBoundary (6.3.7)		

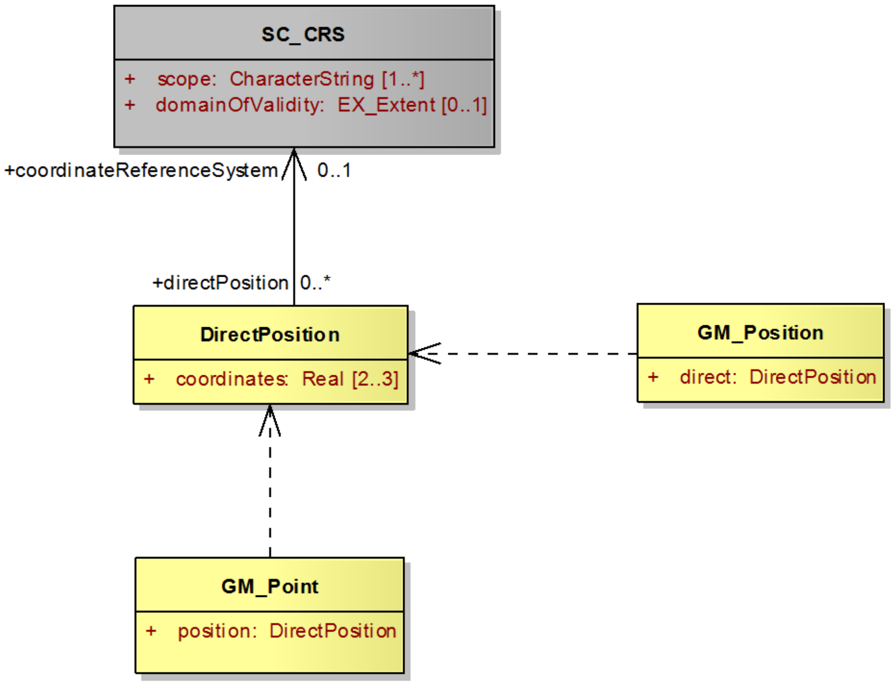

SC_CRS
+ scope: CharacterString [1..*]
+ domainOfValidity: EX_Extent [0..1]
+coordinateReferenceSystem
0..1
+directPosition
0..*
DirectPosition
+ coordinates: Real [2..3]
GM_Position
+ direct: DirectPosition
GM_Point
+ position: DirectPosition

Figure 7-2 — Coordinate Geometry

7-5.1.2 Direct Position

7-5.1.2.1 Semantics

DirectPosition holds the coordinates for a position within a particular coordinate reference system. In this profile, the associated *SC_CRS* must be linked at the *GM_Aggregate* level and not directly to a *DirectPosition*.

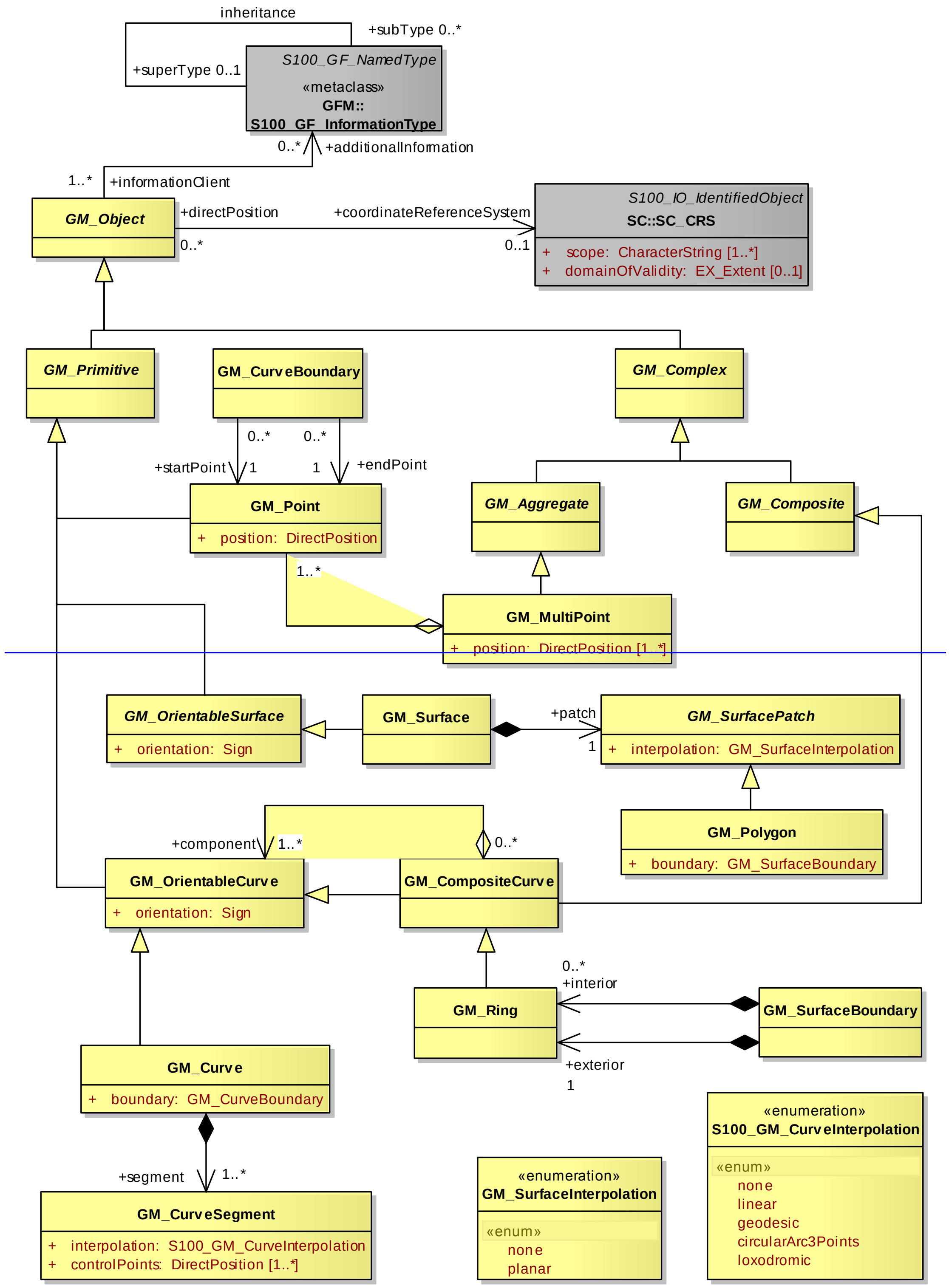
7-5.1.3 GM_Position

7-5.1.3.1 Semantics

The data type *GM_Position* (Figure 7-2) consists of either a *DirectPosition* or a reference to a *GM_Point* (*GM_PointRef*) from which a *DirectPosition* can be obtained.

This profile does not permit the use of the indirect position (*GM_PointRef*).

7-5.2 Simple geometry


inheritance
+subType 0..*
+superType 0..1
S100_GF_NamedType
«metaclass»
GFM::
S100_GF_InformationType
0..*
+additionalInformation
1..*
+informationClient
GM_Object
+directPosition
+coordinateReferenceSystem
0..*
0..1
S100_IO_IdentifiedObject
SC::SC_CRS
+ scope: CharacterString [1..*]
+ domainOfValidity: EX_Extent [0..1]
GM_Primitive
GM_CurveBoundary
GM_Complex
0..*
0..*
+startPoint
1
1
+endPoint
GM_Point
+ position: DirectPosition
GM_Aggregate
GM_Composite
1..*
GM_MultiPoint
+ position: DirectPosition [1..*]
GM_OrientableSurface
+ orientation: Sign
GM_Surface
+patch
1
GM_SurfacePatch
+ interpolation: GM_SurfaceInterpolation
GM_Polygon
+ boundary: GM_SurfaceBoundary
+component
1..*
0..*
GM_OrientableCurve
+ orientation: Sign
GM_CompositeCurve
0..*
+interior
GM_Ring
GM_SurfaceBoundary
+exterior
1
GM_Curve
+ boundary: GM_CurveBoundary
«enumeration»
S100_GM_CurveInterpolation
«enum»
none
linear
geodesic
circularArc3Points
loxodromic
+segment
1..*
GM_CurveSegment
+ interpolation: S100_GM_CurveInterpolation
+ controlPoints: DirectPosition [1..*]
«enumeration»
GM_SurfaceInterpolation
«enum»
none
planar

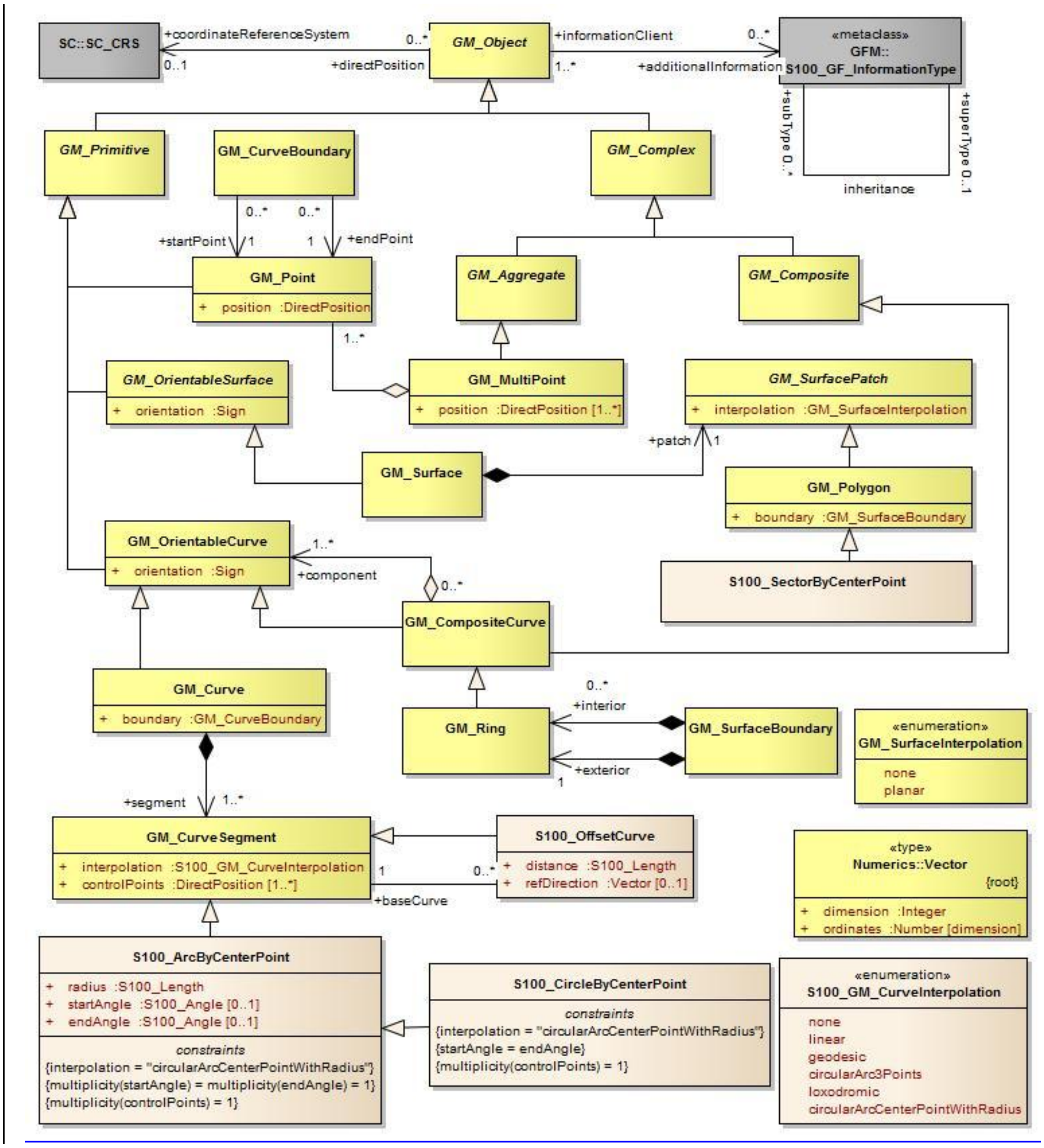


Figure 7-3 — Geometry

7-5.2.1 CurveInterpolation

7-5.2.1.1 Semantics

CurveInterpolation (Figure 7-3) is a list of codes to be used to identify the interpolation mechanisms specified by an application schema.

In this profile, the types of "interpolation" available are limited to the following:

1) Linear (linear) – the interpolation is defined by a series of DirectPositions on a straight line between each consecutive pair of controlPoints.

2) Geodesic (geodesic) – the interpolation mechanism shall return DirectPositions on a geodesic curve between each consecutive pair of controlPoints. A geodesic curve is a curve of shortest length. The geodesic shall be determined in the coordinate reference system of the *GM_Curve* in which the *GM_CurveSegment* is used.

3) Circular arc by 3 points (circularArc3Points) – the interpolation defined by a series of 3 DirectPositions on a circular arc passing from the start point through the middle point to the end point for each set of three consecutive controlPoints. The middle point is located halfway between the start and end point.

4) Loxodromic (loxodromic) – the interpolation method shall return DirectPositions on a loxodromic curve between each consecutive pair of controlPoints. A loxodrome is a line crossing all meridians at the same angle, i.e. a path of constant bearing.

5) Circular arc with centre and radius (circularArcCenterPointWithRadius) – the interpolation is defined by an arc of a circle of the specified radius, centred at the coordinates of the single control point, and drawn from the azimuth of the start angle parameter to the end angle parameter.

7-5.2.2 GM_CurveSegment

7-5.2.2.1 Semantics

A *GM_CurveSegment* (Figure 7-3) defines the position, shape and orientation of a single *GM_Curve*. A *GM_CurveSegment* consists either of positions which are joined by straight lines, or positions which fall on a line defined by a particular type of interpolation as described in 7-5.2.1.

7-5.2.3 SurfaceInterpolation

7-5.2.3.1 Semantics

GM_SurfaceInterpolation (Figure 7-3) is a list of codes which are used to identify the method of interpolation.

In this profile, the types of *interpolation* are constrained to the following:

1) None (none) – the interior of the surface is not specified. The assumption is that the surface follows the reference surface defined by the coordinate reference system.

2) Planar (planar) – the interpolation is a section of a planar, or flat, surface. The boundary in this case shall be contained within that plane.

7-5.2.4 GM_SurfacePatch

7-5.2.4.1 Semantics

The *GM_SurfacePatch* (Figure 7-3) is the abstract root class for all 2-dimensional geometric constructs. It uses a single interpolation to define the shape and position of the associated *GM_Surface* primitives.

7-5.2.5 GM_Polygon

7-5.2.5.1 Semantics

A *GM_Polygon* (Figure 7-3) is defined by a boundary (see 7-5.2.7 below) and an underlying surface to which this boundary is connected. The polygon uses planar interpolation. A *GM_Polygon* is a subtype of *GM_SurfacePatch*.

7-5.2.6 GM_Curve

7-5.2.6.1 Semantics

GM_Curve (Figure 7-3) is a descendent subtype of *GM_Primitive* through *GM_OrientablePrimitive*. It is the basis for 1-dimensional geometry. A curve is a continuous image of an open interval and so could be written as a parameterized function such as c(t):(a, b)□E_n where "t" is a real parameter and En is Euclidean space of dimension n (usually 2 or 3, as determined by the coordinate reference system). Any other parameterization that results in the same image curve, traced in the same direction, such as any linear shifts and positive scales such as e(t) = c(a + t(b-a)):(0,1) □E_n, is an equivalent representation of the same curve. For the sake of simplicity, *GM_Curve* should be parameterized by arc length, so that the parameterization operation inherited from *GM_GenericCurve* (see ISO 19107 clause 6.4.7) will be valid for parameters between 0 and the length of the curve.

Curves are continuous, connected, and have a measurable length in terms of the coordinate system. The orientation of the curve is determined by this parameterization, and is consistent with the tangent function, which approximates the derivative function of the parameterization and shall always point in the "forward" direction. The parameterization of the reversal of the curve defined by c(t):(a, b)□E_n would be defined by a function of the form s(t) = c(a + b - t):(a, b)□E_n.

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.

7-5.2.7 GM_CurveBoundary

7-5.2.7.1 Semantics

The boundary of *GM_Curve* shall be represented as *GM_CurveBoundary*.

7-5.2.8 GM_OrientableCurve

7-5.2.8.1 Semantics

A *GM_OrientableCurve* (Figure 7-3) is a *GM_Curve* with an associated orientation inherited from *GM_OrientablePrimative*.

7-5.2.9 GM_OrientableSurface

7-5.2.9.1 Semantics

A *GM_ OrientableSurface* (Figure 7-3) is a *GM_Surface* with an associated orientation inherited

from its *GM_OrientablePrimative* parent.

7-5.2.10 GM_Point

7-5.2.10.1 Semantics

GM_Point (Figure 7-3) is a 0-dimensional geometric primitive (*GM_Primitive*).

GM_Point is the data type for a geometric object consisting of one and only one point.

7-5.2.11 GM_Primitive

7-5.2.11.1 Semantics

GM_Primitive (Figure 7-3) is the abstract root class for all geometric primitives defined in this profile. A *GM_Primitive* is a *GM_Object*. *GM_Primitive* consists of three sub-types. *GM_Point* which is 0 -dimensional; *GM_Curve* which is 1-dimensional and *GM_Surface* which is 2-dimensional. All geometric primitives (*GM_Primitive*) must be part of at least one *GM_Aggregate* (see ISO 19107 clause 8.10.1). There is no direct link between each *GM_Primitive* and the coordinate reference system *SC_CRS* used for defining the position of the *GM_Primitive.* All *GM_Primitive* contained within a *GM_Aggregate* use the same *SC_CRS* for defining their position.

7-5.2.12 GM_Ring

7-5.2.12.1 Semantics

A *GM_Ring* (Figure 7-3) is composed of a number of references to *GM_OrientableCurves*. The endpoint of *GM_OrientableCurve* “n” is the startPoint of *GM_OrientableCurve* “n+1” and the first startpoint is coincident with the last endpoint, i.e. the *GM_Ring* is closed. A *GM_Ring* must be simple, i.e. it does not intersect itself.

7-5.2.13 GM_Surface

7-5.2.13.1 Semantics

GM_Surface (Figure 7-3) is a subclass of *GM_Primitive* and is the basis for 2-dimensional geometry. It is a *GM_OrientableSurface* with a positive orientation.

This profile does not use instances of *GM_Surface*. A *GM_Surface* within this profile must be subtyped as a *GM_Polygon.*

7-5.2.14 GM_SurfaceBoundary

7-5.2.14.1 Semantics

The boundary of *GM_Surfaces* shall be represented as *GM_SurfaceBoundary* (Figure 7-3).

A *GM_SurfaceBoundary* consists of references to a combination of at least one exterior *GM_Ring* and zero or more interior *GM_Rings*. The rings must be closed as described in ISO 19107 Clause 6.6.11.1.

7-5.2.15 GM_Complex

7-5.2.15.1 Semantics

A *GM_Complex* (Figure 7-3) is a collection of geometrically separate, simple *GM_Primitives*. If a *GM_Primitive* (other than a *GM_Point*) is in a particular *GM_Complex*, then there exists a set of primitives of lower dimension in the same complex that form the boundary of this primitive. For example a *GM_Surface* is a 2 dimensional object, its boundary consists of *GM_Curve* which are 1 dimensional.

7-5.2.16 GM_Composite

7-5.2.16.1 Semantics

A geometric composite, *GM_Composite* (Figure 7-3), is a collection of primitives which must have geometry of the same type and which could exist as a single example of that primitive. For example, a composite curve is a collection of curves which could equally be represented by a single curve. This does not apply to *GM_Point* which can only contain one point.

7-5.2.17 GM_CompositeCurve

7-5.2.17.1 Semantics

A *GM_CompositeCurve* (Figure 7-3) has all the geometric properties of a curve. A composite curve is a sequence of *GM_OrientableCurve*s, each curve (except the first) begins where the previous curve ends.

7-5.2.18 GM_Aggregate

7-5.2.18.1 Semantics

The aggregates, *GM_Aggregates* (Figure 7-3) gather geometric objects. Since they will often use orientation modification, the curve reference and surface references do not go directly to the *GM_Curve* and *GM_Surface*, but are directed to G*M_OrientableCurve* and *GM_OrientableSurface*.

Most geometric objects are contained in features, and cannot be held in collections that are strong aggregations. For this reason, the collections described in this clause are all weak aggregations, and shall use references to include geometric objects.

NOTE The subclasses of *GM_OrientablePrimitive* are handled in such a manner that the reference object can link to a specific orientation of that object.

7-5.2.19 GM_MultiPoint

7-5.2.19.1 Semantics

GM_MultiPoint is an aggregate class containing only points. The association role "element" shall be the set of *GM_Points* contained in this *GM_MultiPoint*.

7-5.2.20 S100_ArcByCenterPoint

7-5.2.20.1 Semantics

An ArcByCenterPoint (Figure 7-4) has the centre of the arc as the single control point plus the radius and bearing of the start and end of the arc.

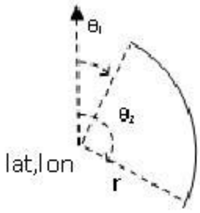


Figure 7-4 – Arc by Centre Point

7-5.2.21 S100_CircleByCenterPoint

7-5.2.21.1 Semantics

An CircleByCenterPoint (Figure 7-5) has the center of the circle as the single control point plus the radius. It is equivalent to an ArcByCenterPoint with identical start and end angle.

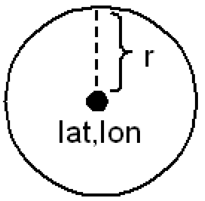


Figure 7-5 – Circle By Centre Point

7-5.2.22 S100_SectorByCenterPoint

7-5.2.22.1 Semantics

A SectorByCenterPoint (Figure 7-6) has the center of the circle and the end points of an arc of the circle as the control points plus the radius and the bearing of the start and end of the sector. It is equivalent to an ArcByCenterPoint combined with the two radii joining the center to the endpoints of the arc.

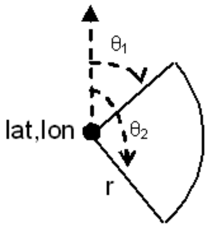


Figure 7-6 – Arc By Centre Point

7-5.2.23 GM_OffsetCurve

7-5.2.23.1 Semantics

An OffsetCurve (Figure 7-7) is a curve at a constant distance from a basis curve. The curve is specified by providing the distance and direction relative to a basis curve. The direction 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.

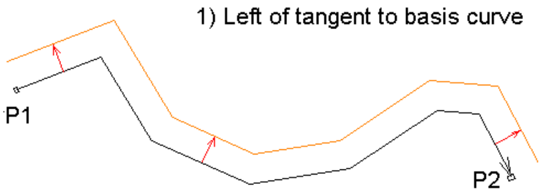


Figure 7-7 – Offset Curve

7-5.3 Geometry configurations

Figure 7-3 depicts a one size fits all geometry model which can be further constrained in both dimensionality and complexity. This is broken down into 5 basic levels (1-3b). Two additional levels (3c and 3d) extend level 3a.

7-5.3.1 Level 1 – 0-, 1-Dimension (no constraints)

A set of isolated point and curve primitives. Curves do not reference points (no boundary), points and curves may be coincident. Areas are represented by a closed loop of curves.

7-5.3.2 Level 2a – 0-, 1-Dimension

A set of point and curve primitives with the following constraints:

1) Each curve must reference a start and end point (they may be the same).
2) Curves must not self intersect.
3) Areas are represented by a closed loop of curves beginning and ending at a common point.
4) In the case of areas with holes, all internal boundaries must be completely contained within the external boundary and the internal boundaries must not intersect each other or the external boundary. Internal boundaries may touch tangentially (i.e. at one point) as shown in Figure 7-8~~4~~a and 7-8~~4~~b.
5) The outer boundary of a surface must be in a clockwise direction (surface to the right of the curve) and the curve orientation positive. The inner boundary of a surface must be in a counter-clockwise direction (surface to the right of the curve) and the curve orientation negative as shown in Figure 7-9~~5~~.

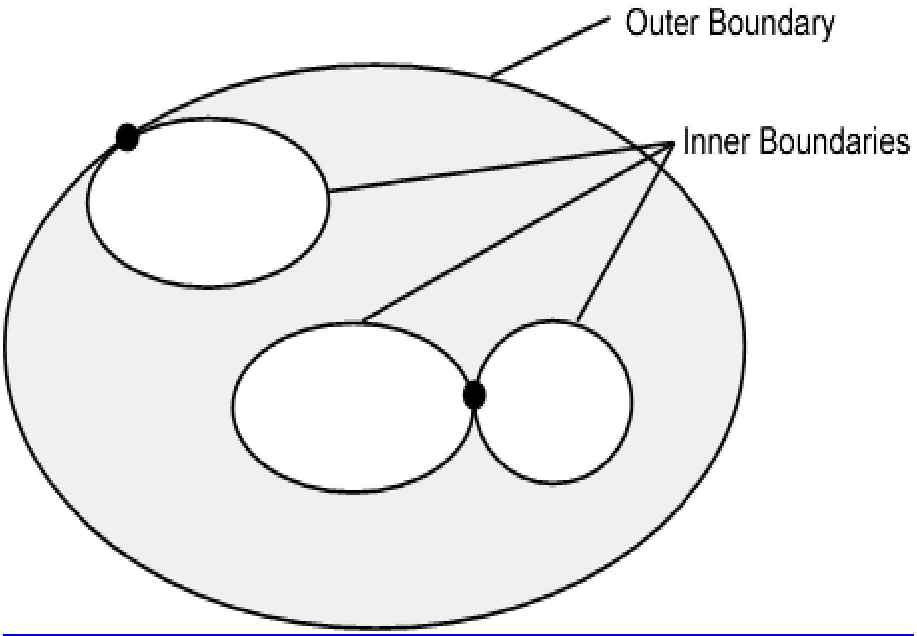


Figure 7-8a — Area Holes

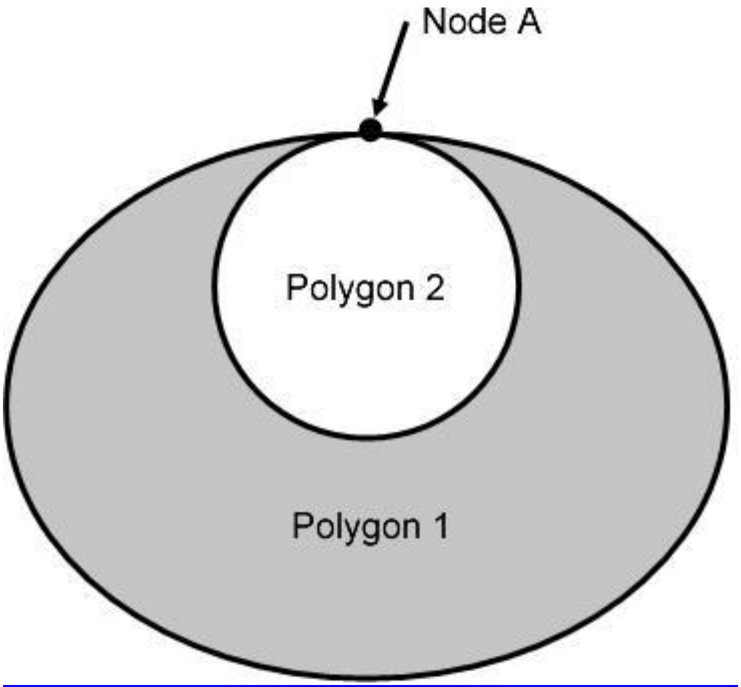


Figure 7-8b — Self – Intersect Example

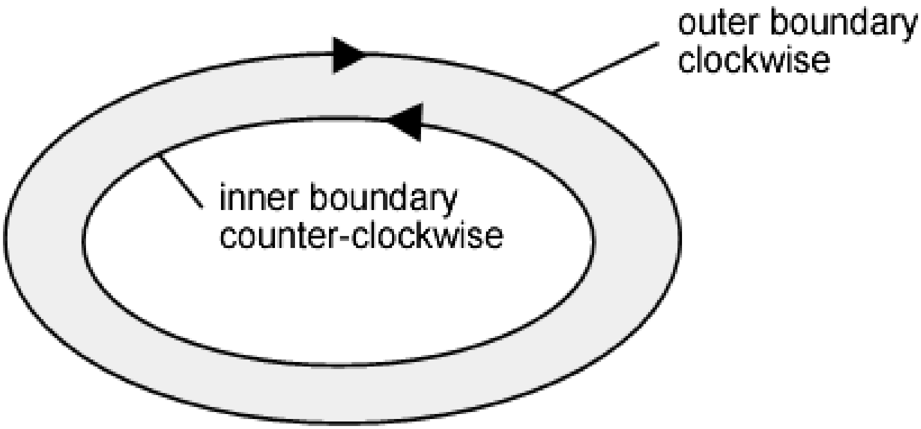


Figure 7-9 — Boundary Direction

7-5.3.3 Level 2b – 0-, 1-Dimension

A set of point and curve primitives. The constraints for Level 2a apply plus the following:

1) Each set of primitives must form a geometric complex.
2) Curves must not intersect without referencing a point at the intersection.
3) Duplication of coincident geometry is prohibited.

7-5.3.4 Level 3a – 0-, 1- and 2-Dimension

A set of point, curve and surface primitives. The constraints for Level 2a applies.

7-5.3.5 Level 3b – 0-, 1- and 2-Dimension

A set of point, curve and surface primitives. The constraints for Levels 2a and 2b apply plus the following:

1) Surfaces must be mutually exclusive and provide exhaustive cover.

7-5.3.6 Level 3c – 0-, 1- and 2-Dimension

Point, curve and surface primitives as for Level 3a, some of which may have attached constraints and/or annotations distinguishing some objects as one of the following four spatial types:

1) S100_ArcByCenterPoint
2) S100_CircleByCenterPoint
3) S100_OffsetCurve
4) S100_SectorByCenterPoint

The constraints for Level 2a apply.

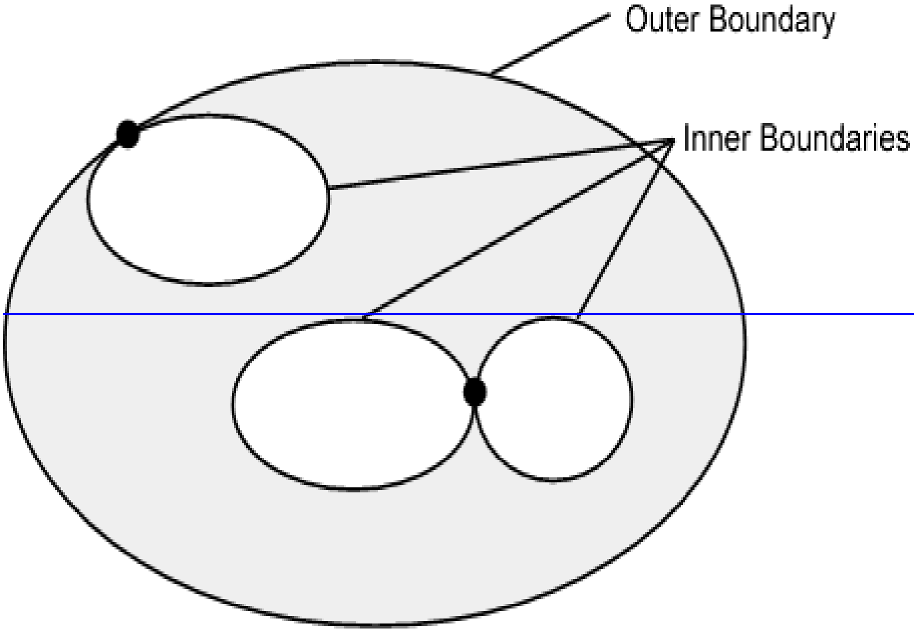
7-5.3.7 Level 3d – 0-, 1- and 2-Dimension

Point, curve, and surface primitives for Level 3a plus the following spatial types:

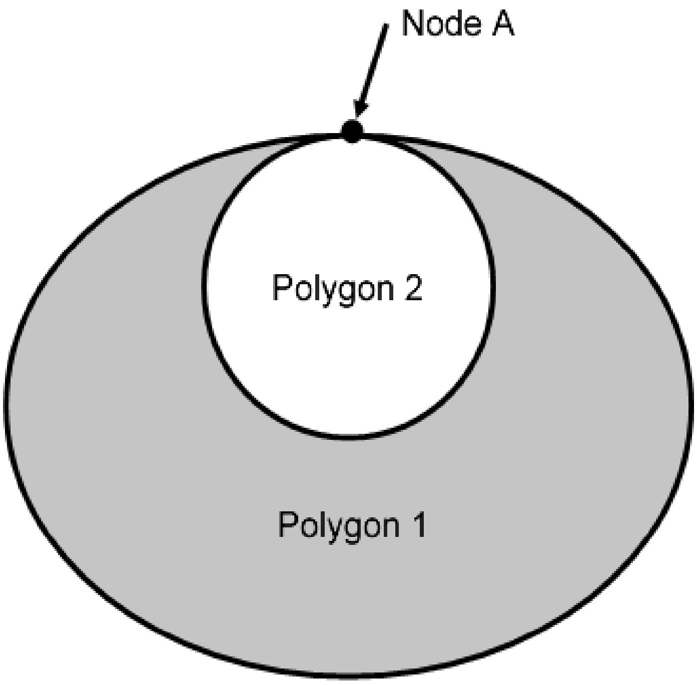
1) S100_ArcByCenterPoint
2) S100_CircleByCenterPoint
3) S100_OffsetCurve
4) S100_SectorByCenterPoint

The constraints for Level 2a apply.

Level 3d objects differ from Level 3c objects in that the four spatial types named in §§ 7-5.3.6 and 7-5.3.7 include explicit modelling of classes and additional attributes representing the added spatial types.

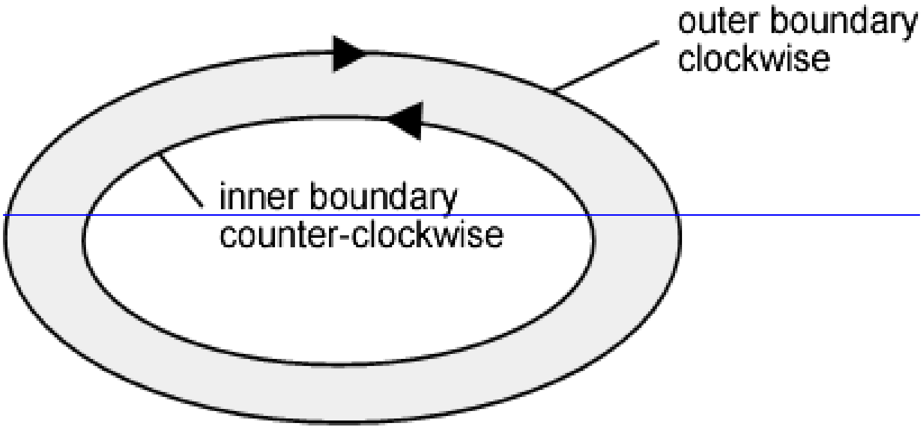


~~Figure 7-4a — Area Holes~~



~~Figure 7-4b — Self - Intersect Example~~

~~Boundary direction~~



~~Figure 7-5 — Boundary Direction~~

Appendix 7-A (informative)

Examples

A.1 Curve Example

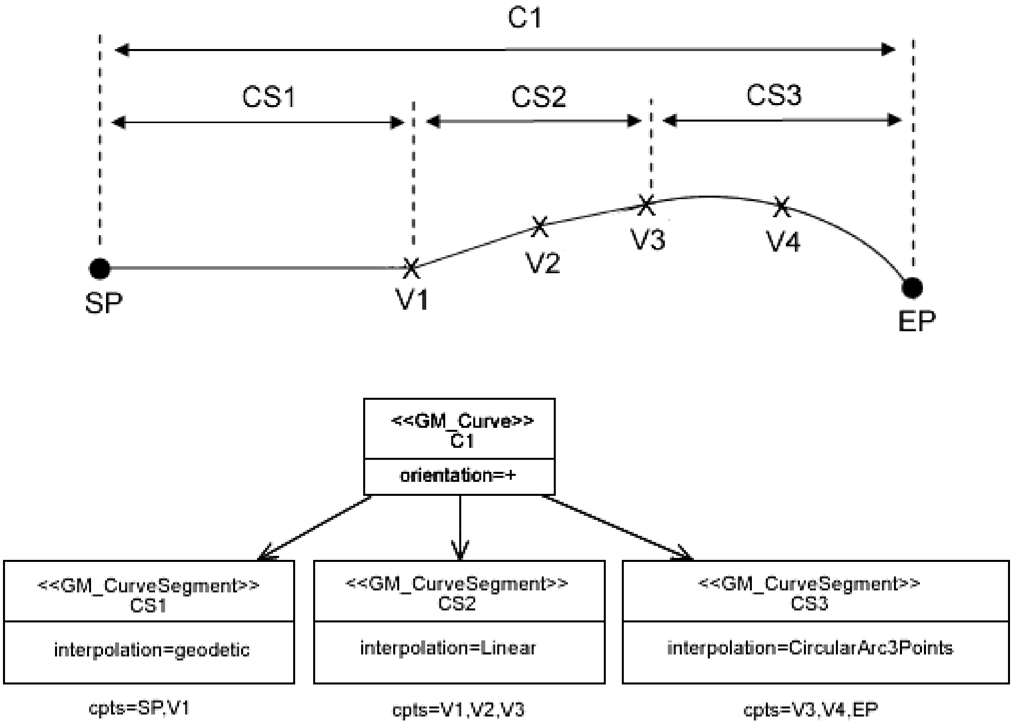


Figure 7-A.1 — Curve Example

The following describes the geometrical elements of the curve example (Figure 7-A.1).

C1 (GM_Curve) consists of CS1, CS2 and CS3 (GM_CurveSegments). CS1 uses a geodetic interpolation, CS2 linear and CS3 arc. SP (start point) and EP (end point) (GM_Points) are the start and end points of C1 and can also be used indirectly as a 0 dimension position for a point feature. An array of control points for each segment consists of a combination of start point, end point and vertices as indicated in the above diagram. The orientation of C1 is + (forward) from SP to EP.

A.2 Surface example

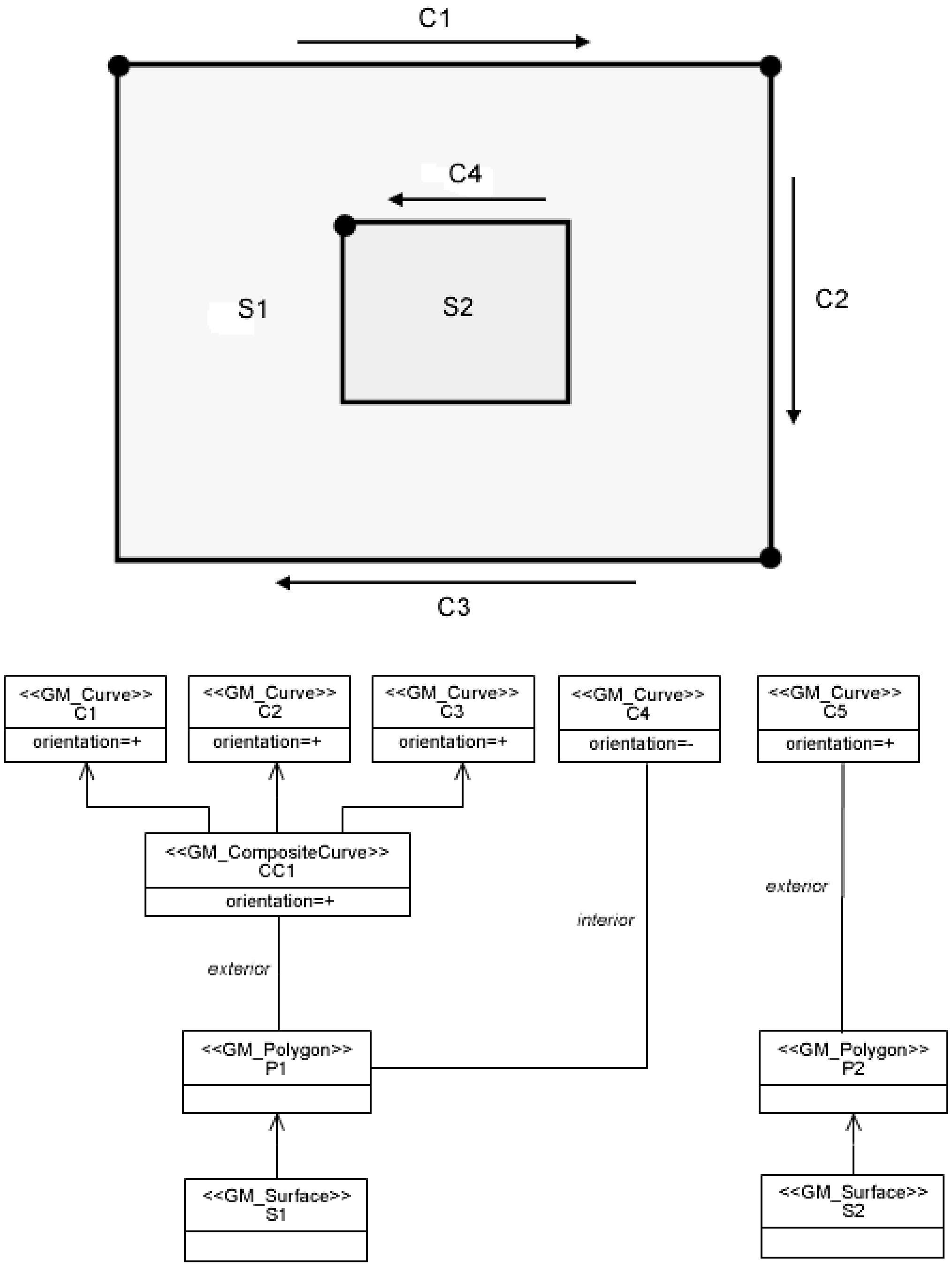


Figure 7-A.2 – Surface Example

The following describes the geometrical elements of the surface example (Figure 7-A.2).

S1 (GM_Surface) is represented by the surface patch P1 (GM_Polygon) the boundary of which consists of exterior and interior rings. The exterior ring CC1 (GM_CompositeCurve) is an aggregation of C1, C2, C3 (GM_Curve), the interior ring C4 is a simple GM_Curve.

A.3 2.5 Dimensional Geometry

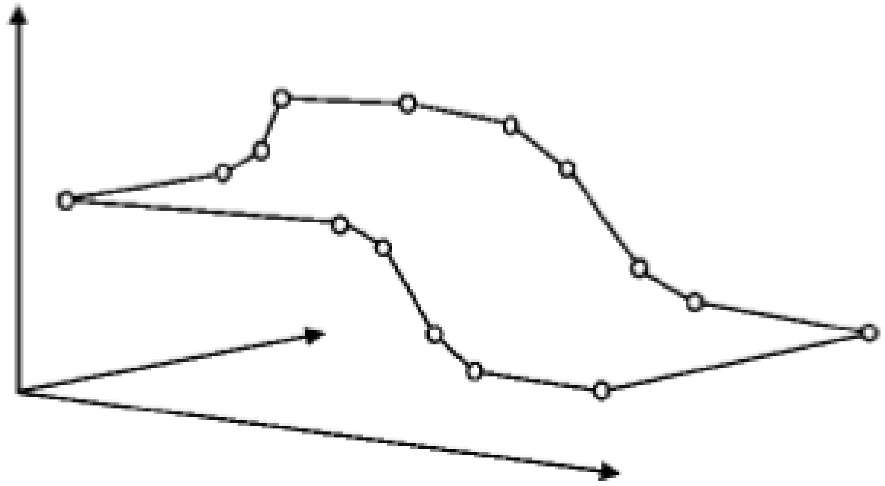


Figure 7-A.3– 2.5D Example

In the depicted example, the curve which constitutes the exterior boundary of a GM_Polygon consists of an array of 3D control points. Note that the surface interpolation must be “none”, which means that the position of interior points is not determined. The “planar” interpolation would only be acceptable if all points were lying on a plane.