INTERNATIONAL STANDARD

ISO 19125-1

First edition
2004-08-01

Corrected version
2004-11-01

Geographic information — Simple feature access —

Part 1: Common architecture

Information géographique — Accès aux entités simples —

Partie 1: Architecture commune

Reference number
ISO 19125-1:2004(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

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.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 19125-1 was prepared by Technical Committee ISO/TC 211, *Geographic information/Geomatics* from a base document supplied by the Open GIS Consortium, Inc.

ISO 19125 consists of the following parts, under the general title *Geographic information — Simple feature access*:

— *Part 1: Common architecture*

— *Part 2: SQL option*

This corrected version of ISO 19125-1:2004 incorporates the following corrections:

— a complete version of Figure 9, which was truncated in the original;

— removal from the Foreword of the reference to ISO 19125-3, which has now been deleted.

Introduction

This part of ISO 19125 describes the common architecture for simple feature geometry. The simple feature geometry object model is Distributed Computing Platform neutral and uses UML notation. The base Geometry class has subclasses for Point, Curve, Surface and GeometryCollection. Each geometric object is associated with a Spatial Reference System, which describes the coordinate space in which the geometric object is defined.

The extended Geometry model has specialized 0, 1 and 2-dimensional collection classes named MultiPoint, MultiLineString and MultiPolygon for modelling geometries corresponding to collections of Points, LineStrings and Polygons, respectively. MultiCurve and MultiSurface are introduced as abstract superclasses that generalize the collection interfaces to handle Curves and Surfaces.

The attributes, methods and assertions for each Geometry class are described in Figure 1 in 6.1.1. In describing methods, *this* is used to refer to the receiver of the method (the object being messaged).

The SFA COM function “signatures” may use a different notation from SFA SQL. COM notation is more familiar for COM programmers. However, UML notation is used throughout this part of ISO 19125. There may also be methods used in this International Standard that differ from one part to another. Where this is the case, the differences are shown within the part.

This part of ISO 19125 implements a profile of the spatial schema described in ISO 19107:2003, *Geographic information — Spatial schema*. Annex A provides a detailed mapping of the schema in this part of ISO 19125 with the schema described in ISO 19107:2003.

Geographic information — Simple feature access —

Part 1:
Common architecture

1 Scope

This part of ISO 19125 establishes a common architecture and defines terms to use within the architecture.

This part of ISO 19125 does not attempt to standardize and does not depend upon any part of the mechanism by which Types are added and maintained, including the following:

a) syntax and functionality provided for defining types;

b) syntax and functionality provided for defining functions;

c) physical storage of type instances in the database;

d) specific terminology used to refer to User Defined Types, for example UDT.

This part of ISO 19125 does standardize names and geometric definitions for Types for Geometry.

This part of ISO 19125 does not place any requirements on how to define the Geometry Types in the internal schema nor does it place any requirements on when or how or who defines the Geometry Types.

2 Conformance

In order to conform to this part of ISO 19125, an implementation shall satisfy the requirements of one or more test suites specified in the other parts of ISO 19125.

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 19107:2003, *Geographic information — Spatial schema*

ISO 19111:2003, *Geographic information — Spatial referencing by coordinates*

4 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

4.1
boundary
set that represents the limit of an entity

NOTE Boundary is most commonly used in the context of geometry, where the set is a collection of points or a collection of objects that represent those points. In other arenas, the term is used metaphorically to describe the transition between an entity and the rest of its domain of discourse.

[ISO 19107]

4.2
buffer
geometric object (4.14) that contains all **direct positions** (4.7) whose distance from a specified geometric object is less than or equal to a given distance

[ISO 19107]

4.3
coordinate
one of a sequence of n-numbers designating the position of a **point** (4.17) in n-dimensional space

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

[adapted from ISO 19111]

4.4
coordinate dimension
number of measurements or axes needed to describe a position in a **coordinate system** (4.6)

[ISO 19107]

4.5
coordinate reference system
coordinate system (4.6) that is related to the real world by a datum

[adapted from ISO 19111]

4.6
coordinate system
set of mathematical rules for specifying how **coordinates** (4.3) are to be assigned to **point** (4.17)

[ISO 19111]

4.7
curve
1-dimensional **geometric primitive** (4.15), 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.

[ISO 19107]

4.7
direct position
position described by a single set of **coordinates** (4.3) within a **coordinate reference system** (4.5)

[ISO 19107]

4.9
end point
last **point** (4.17) of a **curve** (4.7)

[ISO 19107]

4.10
exterior
difference between the universe and the closure

NOTE The concept of exterior is applicable to both topological and geometric complexes.

[ISO 19107]

4.11
feature
abstraction of real world phenomena

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

[adapted from ISO 19101]

4.12
feature attribute
characteristic of a **feature** (4.11)

NOTE A feature attribute has a name, a data type, and a value domain associated to it. A feature attribute for a feature instance also has an attribute value taken from the value domain.

[adapted from ISO 19101]

4.13
geometric complex
set of disjoint **geometric primitives** (4.15) where the **boundary** (4.1) of each geometric primitive can be represented as the union of other geometric primitives of smaller dimension within the same set

NOTE The geometric primitives in the set are disjoint in the sense that no direct position is interior to more than one geometric primitive. The set is closed under boundary operations, meaning that for each element in the geometric complex, there is a collection (also a geometric complex) of geometric primitives that represents the boundary of that element. Recall that the boundary of a point (the only 0D primitive object type in geometry) is empty. Thus, if the largest dimension geometric primitive is a solid (3D), the composition of the boundary operator in this definition terminates after at most 3 steps. It is also the case that the boundary of any object is a cycle.

[ISO 19107]

4.14
geometric object
spatial object representing a geometric set

NOTE A geometric object consists of a geometric primitive, a collection of geometric primitives, or a geometric complex treated as a single entity. A geometric object may be the spatial representation of an object such as a feature or a significant part of a feature.

[ISO 19107]

4.15
geometric primitive
geometric object (4.14) representing a single, connected, homogeneous element of space

NOTE Geometric primitives are non-decomposed objects that represent information about geometric configuration. They include points, curves, surfaces, and solids.

[ISO 19107]

4.16
interior
set of all **direct positions** (4.7) that are on a **geometric object** (4.14) but which are not on its **boundary** (4.1)

NOTE The interior of a topological object is the homomorphic image of the interior of any of its geometric realizations. This is not included as a definition because it follows from a theorem of topology.

[ISO 19107]

4.17
point
0-dimensional **geometric primitive** (4.15), representing a position

NOTE The boundary of a point is the empty set.

[ISO 19107]

4.18
simple feature
feature (4.11) restricted to 2D geometry with linear interpolation between vertices, having both spatial and non spatial attributes

4.19
start point
first **point** (4.17) of a **curve** (4.7)

[ISO 19107]

4.20
surface
2-dimensional **geometric primitive** (4.15), locally representing a continuous image of a region of a plane

NOTE The boundary of a surface is the set of oriented, closed curves that delineate the limits of the surface.

[adapted from ISO 19107]

5 Abbreviated terms

API	Application Program Interface
COM	Component Object Model
CORBA	Common Object Request Broker Architecture
DCE	Distributed Computing Environment
DCOM	Distributed Component Objected Model
DE-9IM	Dimensionally Extended Nine-Intersection Model
IEEE	Institute of Electrical and Electronics Engineers, Inc.
NDR	Little Endian byte order encoding
OLE	Object Linking and Embedding
RPC	Remote Procedure Call
SQL	Structured Query Language

SRID	Spatial Reference System Identifier
XDR	Big Endian byte order encoding
UDT	User Defined Type
UML	Unified Modeling Language
WKB	Well-Known Binary (representation for example, geometry)

6 Architecture

6.1 Geometry object model

6.1.1 Overview

This subclause describes the object model for simple feature geometry. The simple feature geometry object model is Distributed Computing Platform neutral and uses UML notation. The object model for geometry is shown in Figure 1. The base Geometry class has subclasses for Point, Curve, Surface and GeometryCollection. Each geometric object is associated with a Spatial Reference System, which describes the coordinate space in which the geometric object is defined.

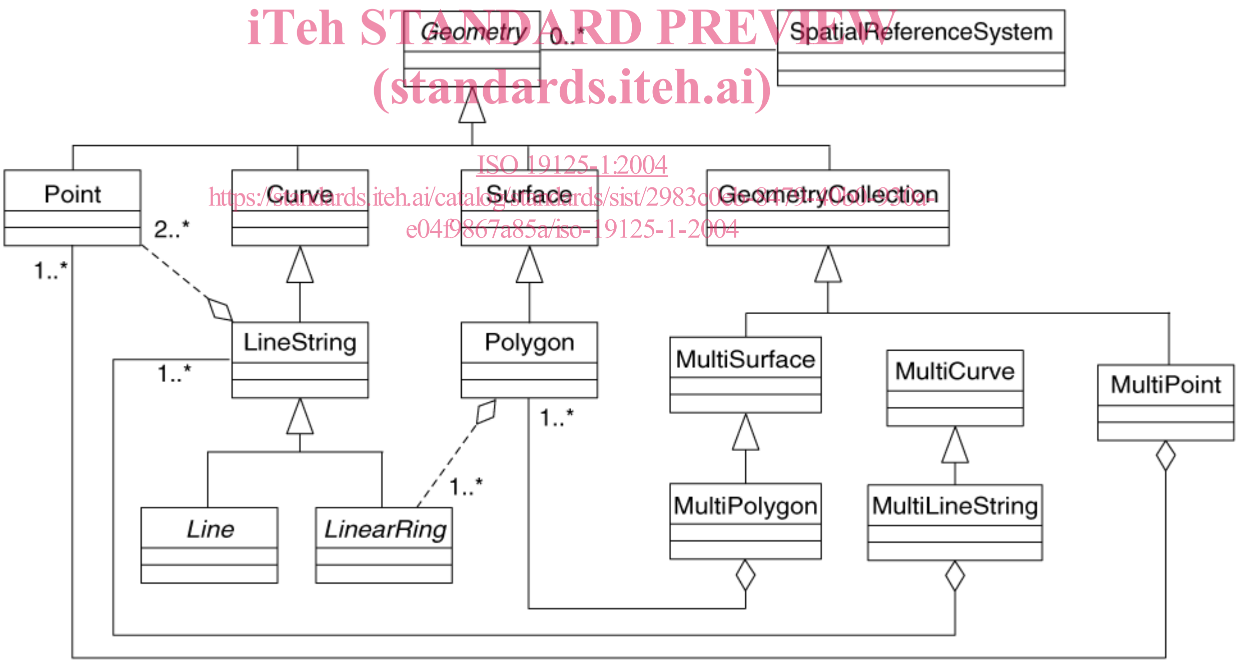


Figure 1 — Geometry class hierarchy

Figure 1 is based on an extended Geometry model with specialized 0-, 1- and 2-dimensional collection classes named MultiPoint, MultiLineString and MultiPolygon for modelling geometries corresponding to collections of Points, LineStrings and Polygons, respectively. MultiCurve and MultiSurface are introduced as abstract superclasses that generalize the collection interfaces to handle Curves and Surfaces. Figure 1 shows aggregation lines between the leaf-collection classes and their element classes; the aggregation lines for non-leaf-collection classes are described in the text.

The attributes, methods and assertions for each Geometry class are described below. In describing methods, *this* is used to refer to the receiver of the method (the object being messaged).

6.1.2 Geometry

6.1.2.1 Description

Geometry is the root class of the hierarchy. Geometry is an abstract (non-instantiable) class.

The instantiable subclasses of Geometry defined in this International Standard are restricted to 0, 1 and 2-dimensional geometric objects that exist in 2-dimensional coordinate space ($\Re^2$).

All instantiable Geometry classes described in this part of ISO 19125 are defined so that valid instances of a Geometry class are topologically closed, i.e. all defined geometries include their boundary.

6.1.2.2 Basic methods on geometric objects

— **Dimension** ( ):Integer — The inherent dimension of *this* geometric object, which must be less than or equal to the coordinate dimension. This specification is restricted to geometries in 2-dimensional coordinate space.

— **GeometryType** ( ):String — Returns the name of the instantiable subtype of Geometry of which *this* geometric object is a instantiable member. The name of the subtype of Geometry is returned as a string.

— **SRID** ( ):Integer — Returns the Spatial Reference System ID for *this* geometric object.

— **Envelope**( ):Geometry — The minimum bounding box for *this* Geometry, returned as a Geometry. The polygon is defined by the corner points of the bounding box [(MINX, MINY), (MAXX, MINY), (MAXX, MAXY), (MINX, MAXY), (MINX, MINY)].

— **AsText**( ):String — Exports *this* geometric object to a specific Well-known Text Representation of Geometry.

— **AsBinary**( ):Binary — Exports *this* geometric object to a specific Well-known Binary Representation of Geometry.

— **IsEmpty**( ):Integer — Returns 1 (TRUE) if *this* geometric object is the empty Geometry. If true, then this geometric object represents the empty point set, ∅, for the coordinate space.

— **IsSimple**( ):Integer — Returns 1 (TRUE) if *this* geometric object has no anomalous geometric points, such as self intersection or self tangency. The description of each instantiable geometric class will include the specific conditions that cause an instance of that class to be classified as not simple.

— **Boundary**( ):Geometry — Returns the closure of the combinatorial boundary of *this* geometric object (Reference [1], section 3.12.2). Because the result of this function is a closure, and hence topologically closed, the resulting boundary can be represented using representational Geometry primitives (Reference [1], section 3.12.2).

6.1.2.3 Methods for testing spatial relations between geometric objects

The methods in this subclause are defined and described in more detail following the description of the sub-types of Geometry.

— **Equals**(anotherGeometry:Geometry):Integer — Returns 1 (TRUE) if *this* geometric object is "spatially equal" to anotherGeometry.

— **Disjoint**(anotherGeometry:Geometry):Integer — Returns 1 (TRUE) if *this* geometric object is "spatially disjoint" from anotherGeometry.

— **Intersects**(anotherGeometry:Geometry):Integer — Returns 1 (TRUE) if *this* geometric object “spatially intersects” anotherGeometry.

— **Touches**(anotherGeometry:Geometry):Integer — Returns 1 (TRUE) if *this* geometric object “spatially touches” anotherGeometry.

— **Crosses**(anotherGeometry:Geometry):Integer — Returns 1 (TRUE) if *this* geometric object “spatially crosses’ anotherGeometry.

— **Within**(anotherGeometry:Geometry):Integer — Returns 1 (TRUE) if *this* geometric object is “spatially within” anotherGeometry.

— **Contains**(anotherGeometry:Geometry):Integer — Returns 1 (TRUE) if *this* geometric object “spatially contains” anotherGeometry.

— **Overlaps**(anotherGeometry:Geometry):Integer — Returns 1 (TRUE) if *this* geometric object “spatially overlaps” anotherGeometry.

— **Relate**(anotherGeometry:Geometry, intersectionPatternMatrix:String):Integer — Returns 1 (TRUE) if *this* geometric object is spatially related to anotherGeometry by testing for intersections between the interior, boundary and exterior of the two geometric objects as specified by the values in the intersectionPatternMatrix.

6.1.2.4 Methods that support spatial analysis

— **Distance**(anotherGeometry:Geometry):Double — Returns the shortest distance between any two Points in the two geometric objects as calculated in the spatial reference system of *this* geometric object.

— **Buffer**(distance:Double):Geometry — Returns a geometric object that represents all Points whose distance from *this* geometric object is less than or equal to distance. Calculations are in the spatial reference system of *this* geometric object.

— **ConvexHull**( ):Geometry — Returns a geometric object that represents the convex hull of *this* geometric object.

— **Intersection**(anotherGeometry:Geometry):Geometry — Returns a geometric object that represents the Point set intersection of *this* geometric object with anotherGeometry.

— **Union**(anotherGeometry:Geometry):Geometry — Returns a geometric object that represents the Point set union of *this* geometric object with anotherGeometry.

— **Difference**(anotherGeometry:Geometry):Geometry — Returns a geometric object that represents the Point set difference of *this* geometric object with anotherGeometry.

— **SymDifference**(anotherGeometry:Geometry):Geometry — Returns a geometric object that represents the Point set symmetric difference of *this* geometric object with anotherGeometry.

6.1.3 GeometryCollection

6.1.3.1 Description

A GeometryCollection is a geometric object that is a collection of 1 or more geometric objects.

All the elements in a GeometryCollection shall be in the same Spatial Reference. This is also the Spatial Reference for the GeometryCollection.

GeometryCollection places no other constraints on its elements. Subclasses of GeometryCollection may restrict membership based on dimension and may also place other constraints on the degree of spatial overlap between elements.

6.1.3.2 Methods

— **NumGeometries**( ):Integer — Returns the number of geometries in *this* GeometryCollection.

— **GeometryN**(N:integer):Geometry — Returns the Nth geometry in *this* GeometryCollection.

6.1.4 Point

6.1.4.1 Description

A Point is a 0-dimensional geometric object and represents a single location in coordinate space. A Point has an x-coordinate value and a y-coordinate value.

The boundary of a Point is the empty set.

6.1.4.2 Methods

— **X**( ):Double — The x-coordinate value for *this* Point.

— **Y**( ):Double — The y-coordinate value for *this* Point.

6.1.5 MultiPoint

A MultiPoint is a 0-dimensional GeometryCollection. The elements of a MultiPoint are restricted to Points. The Points are not connected or ordered.

A MultiPoint is simple if no two Points in the MultiPoint are equal (have identical coordinate values).

The boundary of a MultiPoint is the empty set.

6.1.6 Curve

6.1.6.1 Description

A Curve is a 1-dimensional geometric object usually stored as a sequence of Points, with the subtype of Curve specifying the form of the interpolation between Points. This part of ISO 19125 defines only one subclass of Curve, LineString, which uses linear interpolation between Points.

A Curve is a 1-dimensional geometric object that is the homeomorphic image of a real, closed, interval $D = [a, b] = \{x \in \Re. \ / \ a \leqslant x \leqslant b\}$ under a mapping $f{:}[a,b] \to \Re^2$.

A Curve is simple if it does not pass through the same Point twice (Reference [1], section 3.12.7.3):

$\forall$ c $\in$ Curve, [a, b] = c.Domain,

c.IsSimple$\Leftrightarrow$ ($\forall$ x1, x2 $\in$ (a, b] x1 $\neq$ x2 $\Rightarrow$ f(x1) $\neq$ f (x2)) $\wedge$ ($\forall$ x1, x2 $\in$ [a, b) x1 $\neq$ x2 $\Rightarrow$ f(x1) $\neq$ f(x2))

A Curve is closed if its start Point is equal to its end Point (Reference [1], section 3.12.7.3).

The boundary of a closed Curve is empty.

A Curve that is simple and closed is a Ring.

The boundary of a non-closed Curve consists of its two end Points (Reference [1], section 3.12.3.2).

A Curve is defined as topologically closed.

6.1.6.2 Methods

- **Length**( ):Double — The length of *this* Curve in its associated spatial reference.
- **StartPoint**( ):Point — The start Point of *this* Curve.
- **EndPoint**( ):Point — The end Point of *this* Curve.
- **IsClosed**( ):Integer — Returns 1 (TRUE) if *this* Curve is closed [StartPoint ( ) = EndPoint ( )].
- **IsRing**( ):Integer — Returns 1 (TRUE) if *this* Curve is closed [StartPoint ( ) = EndPoint ( )] and *this* Curve is simple (does not pass through the same Point more than once).

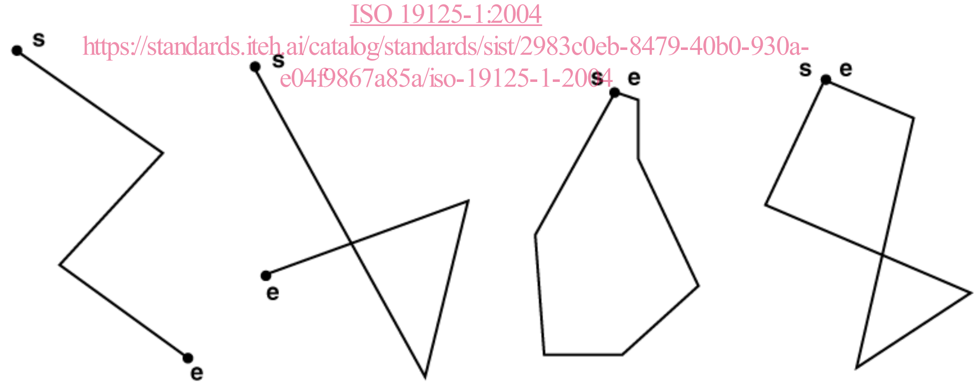
6.1.7 LineString, Line, LinearRing

6.1.7.1 Description

A LineString is a Curve with linear interpolation between Points. Each consecutive pair of Points defines a Line segment.

A Line is a LineString with exactly 2 Points.

A LinearRing is a LineString that is both closed and simple. The Curve in Figure 2, item (c), is a closed LineString that is a LinearRing. The Curve in Figure 2, item (d) is a closed LineString that is not a LinearRing.



a) b) c) d)

Key

s start

e end

Figure 2 — Examples of LineStrings — Simple LineString (a), Non-simple LineString (b), Simple, closed LineString (a LinearRing) (c), Non-simple closed LineString (d)

6.1.7.2 Methods

- **NumPoints**( ):Integer — The number of Points in *this* LineString.