INTERNATIONAL STANDARD

ISO 19123

First edition
2005-08-15

Geographic information — Schema for coverage geometry and functions

Information géographique — Schéma de la géométrie et des fonctions de couverture

Reference number
ISO 19123:2005(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 19123 was prepared by Technical Committee ISO/TC 211, *Geographic information/Geomatics*.

Introduction

Geographic phenomena fall into two broad categories — discrete and continuous. Discrete phenomena are recognizable objects that have relatively well-defined boundaries or spatial extent. Examples include buildings, streams and measurement stations. Continuous phenomena vary over space and have no specific extent. Examples include temperature, soil composition and elevation. A value or description of a continuous phenomenon is only meaningful at a particular position in space (and possibly time). Temperature, for example, takes on specific values only at defined locations, whether measured or interpolated from other locations.

These concepts are not mutually exclusive. In fact, many components of the landscape may be viewed alternatively as discrete or continuous. For example, a stream is a discrete entity, but its flow rate and water quality index vary from one position to another. Similarly, a highway can be thought of as a feature or as a collection of observations measuring accidents or traffic flow, and an agricultural field is both a spatial object and a set of measurements of crop yield through time.

Historically, geographic information has been treated in terms of two fundamental types called vector data and raster data.

“Vector data” deals with discrete phenomena, each of which is conceived of as a feature. The spatial characteristics of a discrete real-world phenomenon are represented by a set of one or more geometric primitives (points, curves, surfaces or solids). Other characteristics of the phenomenon are recorded as feature attributes. Usually, a single feature is associated with a single set of attribute values. ISO 19107:2003 provides a schema for describing features in terms of geometric and topological primitives.

“Raster data”, on the other hand, deals with real-world phenomena that vary continuously over space. It contains a set of values, each associated with one of the elements in a regular array of points or cells. It is usually associated with a method for interpolating values at spatial positions between the points or within the cells. Since this data structure is not the only one that can be used to represent phenomena that vary continuously over space, this International Standard uses the term “coverage,” adopted from the Abstract Specification of the Open GIS Consortium [1], to refer to any data representation that assigns values directly to spatial position. A coverage is a function from a spatial, temporal or spatiotemporal domain to an attribute range. A coverage associates a position within its domain to a record of values of defined data types.

In this International Standard, coverage is a subtype of feature. A coverage is a feature that has multiple values for each attribute type, where each direct position within the geometric representation of the feature has a single value for each attribute type.

Just as the concepts of discrete and continuous phenomena are not mutually exclusive, their representations as discrete features or coverages are not mutually exclusive. The same phenomenon may be represented as either a discrete feature or a coverage. A city may be viewed as a discrete feature that returns a single value for each attribute, such as its name, area and total population. The city feature may also be represented as a coverage that returns values such as population density, land value or air quality index for each position in the city.

A coverage, moreover, can be derived from a collection of discrete features with common attributes, the values of the coverage at each position being the values of the attributes of the feature located at that position. Conversely, a collection of discrete features can be derived from a coverage, each discrete feature being composed of a set of positions associated with specified attribute values.

Geographic information — Schema for coverage geometry and functions

1 Scope

This International Standard defines a conceptual schema for the spatial characteristics of coverages. Coverages support mapping from a spatial, temporal or spatiotemporal domain to feature attribute values where feature attribute types are common to all geographic positions within the domain. A coverage domain consists of a collection of direct positions in a coordinate space that may be defined in terms of up to three spatial dimensions as well as a temporal dimension. 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 mapping of bathymetry, elevation, soil and vegetation. This International Standard defines the relationship between the domain of a coverage and an associated attribute range. The characteristics of the spatial domain are defined whereas the characteristics of the attribute range are not part of this standard.

2 Conformance

This International Standard specifies interfaces for several types of coverage objects. In addition, it supports the interchange of coverage data independently of those interfaces. Thus, it specifies two sets of conformance classes: one for implementation of the interfaces, the other for the exchange of coverage data. Each set includes one conformance class for each type of coverage specified in this International Standard (Table 1).

Table 1 — Conformance classes

Conformance class	Subclause
Simple coverage interface	A.1.1
Discrete coverage interface	A.1.2
Thiessen polygon coverage interface	A.1.3
Quadrilateral grid coverage interface	A.1.4
Hexagonal grid coverage interface	A.1.5
TIN coverage interface	A.1.6
Segmented curve coverage interface	A.1.7
Discrete coverage interchange	A.2.1
Thiessen polygon coverage interchange	A.2.2
Quadrilateral grid coverage interchange	A.2.3
Hexagonal grid coverage interchange	A.2.4
TIN coverage interchange	A.2.5
Segmented curve coverage interchange	A.2.6

In general, the interface conformance classes require implementation of all attributes, associations and operations of relevant classes. This set includes a single conformance class (A.2.1) that supports a simple interface for evaluation of any coverage type, but exposes none of the internal structure of the coverage. The remainder of the set are conformance classes that support interfaces to specific coverage types that expose additional information about the internal structure of the coverage.

The interchange conformance classes require only implementation of the attributes and associations of the relevant classes.

The Abstract Test Suite in Annex A shows the implementation requirements necessary to conform to this International Standard. Table 1 lists the subclauses of the Abstract Test Suite that apply for each conformance class.

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/TS 19103:2005, *Geographic information — Conceptual schema language*

ISO 19107:2003, *Geographic information — Spatial schema*

ISO 19108:2002, *Geographic information — Temporal schema*

ISO 19109:2005, *Geographic information — Rules for application schema*

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

ISO 19115:2003, *Geographic information — Metadata*

4 Terms, definitions, abbreviated terms and notation

4.1 Terms and definitions

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

4.1.1
continuous coverage
coverage that returns different values for the same feature attribute at different **direct positions** within a single **spatial object**, **temporal object** or **spatiotemporal object** in its **domain**

NOTE Although the domain of a continuous coverage is ordinarily bounded in terms of its spatial and/or temporal extent, it can be subdivided into an infinite number of direct positions.

4.1.2
convex hull
smallest **convex set** containing a given **geometric object**

[adapted from *Dictionary of Computing*:1996 [2]]

4.1.3
convex set
geometric set in which any **direct position** on the straight-line segment joining any two **direct positions** in the **geometric set** is also contained in the **geometric set**

[*Dictionary of Computing*:1996 [2]]

4.1.4
coordinate
one of a sequence of *n* numbers designating the position of a **point** in *n*-dimensional space

[ISO 19111:2003]

4.1.5
coordinate dimension
number of measurements or axes needed to describe a position in a coordinate system

[ISO 19107:2003]

4.1.6
coordinate reference system
coordinate system that is related to the real world by a datum

[ISO 19111:2003]

41.7
coverage
feature that acts as a **function** to return values from its **range** for any **direct position** within its spatial, temporal or **spatiotemporal domain**

EXAMPLE Examples include a raster image, polygon overlay or digital elevation matrix.

NOTE In other words, a coverage is a feature that has multiple values for each attribute type, where each direct position within the geometric representation of the feature has a single value for each attribute type.

4.1.8
coverage geometry
configuration of the **domain** of a **coverage** described in terms of **coordinates**

4.1.9
curve
1-dimensional **geometric primitive**, representing the continuous image of a line

[ISO 19107:2003]

NOTE The boundary of a curve is the set of points at either end of the curve.

4.1.10
Delaunay triangulation
network of triangles such that the circle passing through the vertices of any triangle does not contain, in its interior, the vertex of any other triangle

4.1.11
direct position
position described by a single set of **coordinates** within a **coordinate reference system**

[ISO 19107:2003]

4.1.12
discrete coverage
coverage that returns the same **feature attribute** values for every **direct position** within any single **spatial object**, **temporal object** or **spatiotemporal object** in its **domain**

NOTE The domain of a discrete coverage consists of a finite set of spatial, temporal, or spatiotemporal objects.

4.1.13
domain
well-defined set

[ISO/TS 19103]

NOTE Domains are used to define the domain and range of operators and functions.

4.1.14
evaluation
〈**coverage**〉 determination of the values of a **coverage** at a **direct position** within the **domain** of the coverage

4.1.15
feature
0 abstraction of real world phenomena

[ISO 19101]

4.1.16
feature attribute
characteristic of a **feature**

[ISO 19101]

4.1.17
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 19107:2003]

4.1.18
geometric object
spatial object representing a **geometric set**

[ISO 19107:2003]

4.1.19
geometric primitive
geometric object representing a single, connected, homogeneous element of space

[ISO 19107:2003]

4.1.20
geometric set
set of **direct positions**

[ISO 19107:2003]

4.1.21
geometry value object
object composed of a set of **geometry value pairs**

4.1.22
geometry value pair
ordered pair composed of a **spatial object**, a temporal object or a **spatiotemporal object** and a **record** of **feature attribute values**

4.1.23
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

NOTE The curves partition a space into grid cells.

4.1.24
grid point
point located at the intersection of two or more **curves** in a **grid**

4.1.25
inverse evaluation
⟨coverage⟩ selection of a set of objects from the **domain** of a **coverage** based on the **feature attribute** values associated with the objects

4.1.26
point
0-dimensional **geometric primitive**, representing a position

[ISO 19107:2003]

NOTE The boundary of a point is the empty set.

4.1.27
point coverage
coverage that has a **domain** composed of **points**

4.1.28
polygon coverage
coverage that has a **domain** composed of polygons

4.1.29
range
⟨coverage⟩ set of **feature attribute** values associated by a **function** with the elements of the **domain** of a **coverage**

4.1.30
raster
usually rectangular pattern of parallel scanning lines forming or corresponding to the display on a cathode ray tube

NOTE A raster is a type of grid.

4.1.31
record
finite, named collection of related items (objects or values)

[ISO 19107:2003]

NOTE Logically, a record is a set of pairs <name, item>.

4.1.32
rectified grid
grid for which there is an affine transformation between the grid **coordinates** and the coordinates of an external **coordinate reference system**

NOTE If the coordinate reference system is related to the earth by a datum, the grid is a georectified grid.

4.1.33
referenceable grid
grid associated with a transformation that can be used to convert grid **coordinate** values to values of coordinates referenced to an external **coordinate reference system**

NOTE If the coordinate reference system is related to the earth by a datum, the grid is a georeferenceable grid.

4.1.34
solid
3-dimensional **geometric primitive**, representing the continuous image of a region of Euclidean 3-space

[ISO 19107:2003]

NOTE A solid is realizable locally as a three-parameter set of direct positions. The boundary of a solid is the set of oriented, closed surfaces that comprise the limits of the solid.

4.1.35
spatial object
object used for representing a spatial characteristic of a **feature**

[ISO 19107:2003]

4.1.36
spatiotemporal domain
⟨coverage⟩ **domain** composed of **spatiotemporal objects**

NOTE The spatiotemporal domain of a continuous coverage consists of a set of direct positions defined in relation to a collection of spatiotemporal objects.

4.1.37
spatiotemporal object
object representing a set of **direct positions** in space and time

4.1.38
surface
2-dimensional **geometric primitive**, locally representing a continuous image of a region of a plane

[ISO 19107:2003]

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

4.1.39
tessellation
partitioning of a space into a set of conterminous subspaces having the same dimension as the space being partitioned

NOTE A tessellation composed of congruent regular polygons or polyhedra is a regular tessellation. One composed of regular, but non-congruent polygons or polyhedra is a semi-regular tessellation. Otherwise, the tessellation is irregular.

EXAMPLES Graphic examples of tessellations may be found in Figures 11, 13, 20 and 22 of this International Standard.

4.1.40
Thiessen polygon
polygon that encloses one of a set of **points** on a plane so as to include all **direct positions** that are closer to that point than to any other point in the set

4.1.41
topological dimension
minimum number of free variables needed to distinguish nearby **direct positions** within a **geometric object** from one another

[ISO 19107:2003]

4.1.42
triangulated irregular network
tessellation composed of triangles

4.1.43
vector
quantity having direction as well as magnitude

NOTE A directed line segment represents a vector if the length and direction of the line segment are equal to the magnitude and direction of the vector. The term vector data refers to data that represents the spatial configuration of features as a set of directed line segments.

4.2 Abbreviated terms

GIS Geographic Information System

TIN Triangulated Irregular Network

UML Unified Modelling Language

4.3 Notation

The conceptual schema specified in this International Standard is described using the Unified Modelling Language (UML) [4], following the guidance of ISO/TS 19103. Annex B describes UML notation as used in this International Standard.

Several model elements used in this schema are defined in other International Standards developed by ISO/TC 211. By convention within ISO/TC 211, names of UML classes, with the exception of basic data type classes, include a two-letter prefix that identifies the standard and the UML package in which the class is defined. UML classes defined in this International Standard have the two-letter prefix of CV. Table 2 lists the other standards and packages in which UML classes used in this International Standard have been defined.

Table 2 — Sources of externally defined UML classes

Prefix	International Standard	Package
EX	ISO 19115	Extent
GF	ISO 19109	General Feature Model
GM	ISO 19107	Geometry
SC	ISO 19111	Spatial Coordinates
TM	ISO 19108	Temporal Schema

5 Fundamental characteristics of coverages

5.1 The context for coverages

5.1.1 General

A coverage is a feature that associates positions within a bounded space (its domain) to feature attribute values (its range). In other words, it is both a feature and a function. Examples include a raster image, a polygon overlay or a digital elevation matrix.

A coverage may represent a single feature or a set of features.

5.1.2 Domain of a coverage

A coverage domain is a set of geometric objects described in terms of direct positions. It may be extended to all of the direct positions within the convex hull of that set of geometric objects. The direct positions are associated with a spatial or temporal coordinate reference system. Commonly used domains include point sets, grids, collections of closed rectangles, and other collections of geometric objects. The geometric objects may exhaustively partition the domain, and thereby form a tessellation such as a grid or a TIN. Point sets and other sets of non-conterminous geometric objects do not form tessellations. Coverage subtypes may be defined in terms of their domains.

Coverage domains differ in both the coordinate dimension of the space in which they exist and in the topological dimension of the geometric objects they contain. Clearly, the geometric objects that make up a domain cannot have a topological dimension greater than the coordinate dimension of the domain. A domain of coordinate dimension 3 may be composed of points, curves, surfaces, or solids, while a domain of coordinate dimension 2 may be composed only of points, curves or surfaces. ISO 19107:2003 defines a number of geometric objects (subtypes of the UML class GM_Object) to be used for the description of features. Many of these geometric objects can be used to define domains for coverages. In addition, ISO 19108:2002 defines TM_GeometricPrimitives that may also be used to define domains of coverages.

Generally, the geometric objects that make up the domain of a coverage are disjoint, but this International Standard does allow a coverage domain to contain overlapping geometric objects.

5.1.3 The range of a coverage

The range of a coverage is a set of feature attribute values. It may be either a finite or a transfinite set. Coverages often model many associated functions sharing the same domain. Therefore, the value set is represented as a collection of records with a common schema.

EXAMPLE A coverage might assign to each direct position in a county the temperature, pressure, humidity, and wind velocity at noon, today, at that point. The coverage maps every direct position in the county to a record of four fields.

A feature attribute value may be of any data type. However, evaluation of a continuous coverage is usually implemented by interpolation methods that can be applied only to numbers or vectors. Other data types are almost always associated with discrete coverages.

Given a record from the range of a coverage, inverse evaluation is the calculation and exposure of a set of geometric objects associated with specific values of the attributes. Inverse evaluation may return many geometric objects associated with a single feature attribute value.

EXAMPLE Inverse evaluation is used for the extraction of contours from an elevation coverage and the extraction of classified regions in an image.

5.1.4 Discrete and continuous coverages

Coverages are of two types. A discrete coverage has a domain that consists of a finite collection of geometric objects and the direct positions contained in those geometric objects. A discrete coverage maps each geometric object to a single record of feature attribute values. The geometric object and its associated record form a geometry value pair. A discrete coverage is thus a discrete or step function as opposed to a continuous coverage. Discrete functions can be explicitly enumerated as (input, output) pairs. A discrete coverage may be represented as a collection of ordered pairs of independent and dependent variables. Each independent variable is a geometric object and each dependent variable is a record of feature attribute values.

EXAMPLE A coverage that maps a set of polygons to the soil type found within each polygon is an example of a discrete coverage.

A continuous coverage has a domain that consists of a set of direct positions in a coordinate space. A continuous coverage maps direct positions to value records.

EXAMPLE Consider a coverage that maps direct positions in San Diego County to their temperature at noon today. Both the domain and the range may take an infinite number of different values. This continuous coverage would be associated with a discrete coverage that holds the temperature values observed at a set of weather stations.

A continuous coverage may consist of no more than a spatially bounded, but transfinite set of direct positions, and a mathematical function that relates direct position to feature attribute value. This is called an analytical coverage.

EXAMPLE A statistical trend surface that relates land value to position relative to a city centre is an example of a continuous coverage.

More often, the domain of a continuous coverage consists of the direct positions in the union or in the convex hull of a finite collection of geometric objects; it is specified by that collection. In most cases, a continuous coverage is also associated with a discrete coverage that provides a set of control values to be used as a basis for evaluating the continuous coverage. Evaluation of the continuous coverage at other direct positions is done by interpolating between the geometry value pairs of the control set. This often depends upon additional geometric objects constructed from those in the control set; these additional objects are typically of higher topological dimension than the control objects. In this International Standard, such objects are called geometry value objects. A geometry value object is a geometric object associated with a set of geometry value pairs that provide the control for constructing the geometric object and for evaluating the coverage at direct positions within the geometric object.

EXAMPLE Evaluation of a triangulated irregular network involves interpolation of values within a triangle composed of three neighbouring point value pairs.

5.2 The coverage schema

The coverage schema is organized into seven packages with the inter-package dependencies shown in Figure 1. The Coverage Core package is documented in this clause, and each of the other packages is described in a separate clause as shown in Table 3.

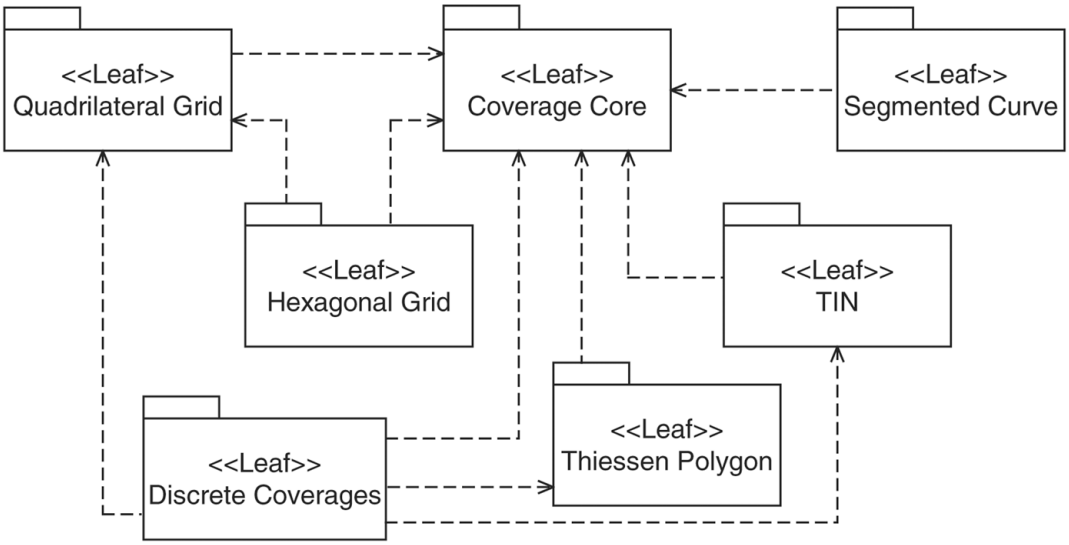


Figure 1 — Packages of the coverage schema

Table 3 — Documentation of coverage geometry packages

Package	Clause
Coverage core	5
Discrete coverages	6
Thiessen polygon	7
Quadrilateral grid	8
Hexagonal grid	9
TIN	10
Segmented curve	11

5.3 CV_Coverage

5.3.1 General

The class CV_Coverage (Figure 2) is an instance of the <<metaclass>> GF_FeatureType (ISO 19109), which therefore represents a feature type. CV_Coverage shall support three attributes, five operations, and three associations.

5.3.2 domainExtent

The attribute *domainExtent[1..*]EX_Extent* shall contain the extent of the domain of the coverage. The data type EX_Extent is defined in ISO 19115:2003. Extents may be specified in space, time or space-time.

5.3.3 rangeType

The attribute *rangeType: RecordType* shall describe the range of the coverage. The data type RecordType is defined in ISO/TS 19103. It consists of a list of attribute name/data type pairs. A simple list is the most common form of *rangeType*, but RecordType can be used recursively to describe more complex structures. The *rangeType* for a specific coverage shall be specified in an application schema.

5.3.4 commonPointRule

The attribute *commonPointRule: CV_CommonPointRule* shall identify the procedure to be used for evaluating the CV_Coverage at a position that falls either on a boundary between geometric objects or within the boundaries of two or more overlapping geometric objects, where the geometric objects are either CV_DomainObjects or CV_ValueObjects. The data type CV_CommonPointRule is defined in 5.6.

5.3.5 list

The operation *list(): Set <CV_GeometryValuePair>* shall return the dictionary of CV_GeometryValuePairs (5.8) that contain the CV_DomainObjects in the domain of the CV_Coverage each paired with its record of feature attribute values. In the case of an analytical coverage, the operation shall return the empty set.

5.3.6 select

The operation *select (s: GM_Object, t: TM_Period): Set <CV_GeometryValuePair>* shall accept a GM_Object and a TM_Period as input and return the set of CV_GeometryValuePairs that contain CV_DomainObjects that lie within that GM_Object and TM_Period. If s is null, the operation shall return all CV_GeometryValuePairs that contain CV_DomainObjects within t. If the value of t is null, the operation shall return all CV_GeometryValuePairs that contain CV_DomainObjects within s. In the case of an analytical coverage, the operation shall return the empty set.

5.3.7 find

The operation *find (p: DirectPosition, limit: Integer = 1): Sequence <CV_GeometryValuePair>* shall accept a DirectPosition as input and return the sequence of CV_GeometryValuePairs that includes the CV_DomainObjects nearest to the DirectPosition and their distances from the DirectionPosition. The sequence shall be ordered by distance from the DirectPosition, beginning with the Record containing the CV_DomainObject nearest to the DirectPosition. The length of the sequence (the number of CV_GeometryValuePairs returned) shall be no greater than the number specified by the parameter *limit*. The default shall be to return a single CV_GeometryValuePair. The operation shall return a warning if the last CV_DomainObject in the sequence is at a distance from the DirectPosition equal to the distance of other CV_DomainObjects that are not included in the sequence. In the case of an analytical coverage, the operation shall return the empty set.

NOTE This operation is useful when the domain of a coverage does not exhaustively partition the extent of the coverage. Even in that case, the first element of the sequence returned may be the CV_GeometryValuePair that contains the input DirectPosition.

5.3.8 evaluate

The operation *evaluate(p: DirectPosition, list: Sequence <CharacterString>): Set <Record>* shall accept a DirectPosition as input and return a set of Records of feature attribute values for that direct position. The parameter *list* is a sequence of feature attribute names each of which identifies a field of the *rangeType*. If *list* is null, the operation shall return a value for every field of the *rangeType*. Otherwise, it shall return a value for each field included in *list*. The data type DirectPosition is defined in ISO 19107:2003; the data type Record is defined in ISO/TS 19103. If the direct position passed is not in the domain of the coverage, then an error message shall be generated. If the input DirectPosition falls within two or more geometric objects within the domain, the operation shall return records of feature attribute values computed according to the value of the attribute *commonPointRule*.

NOTE Normally, the operation will return a single record of feature attribute values.

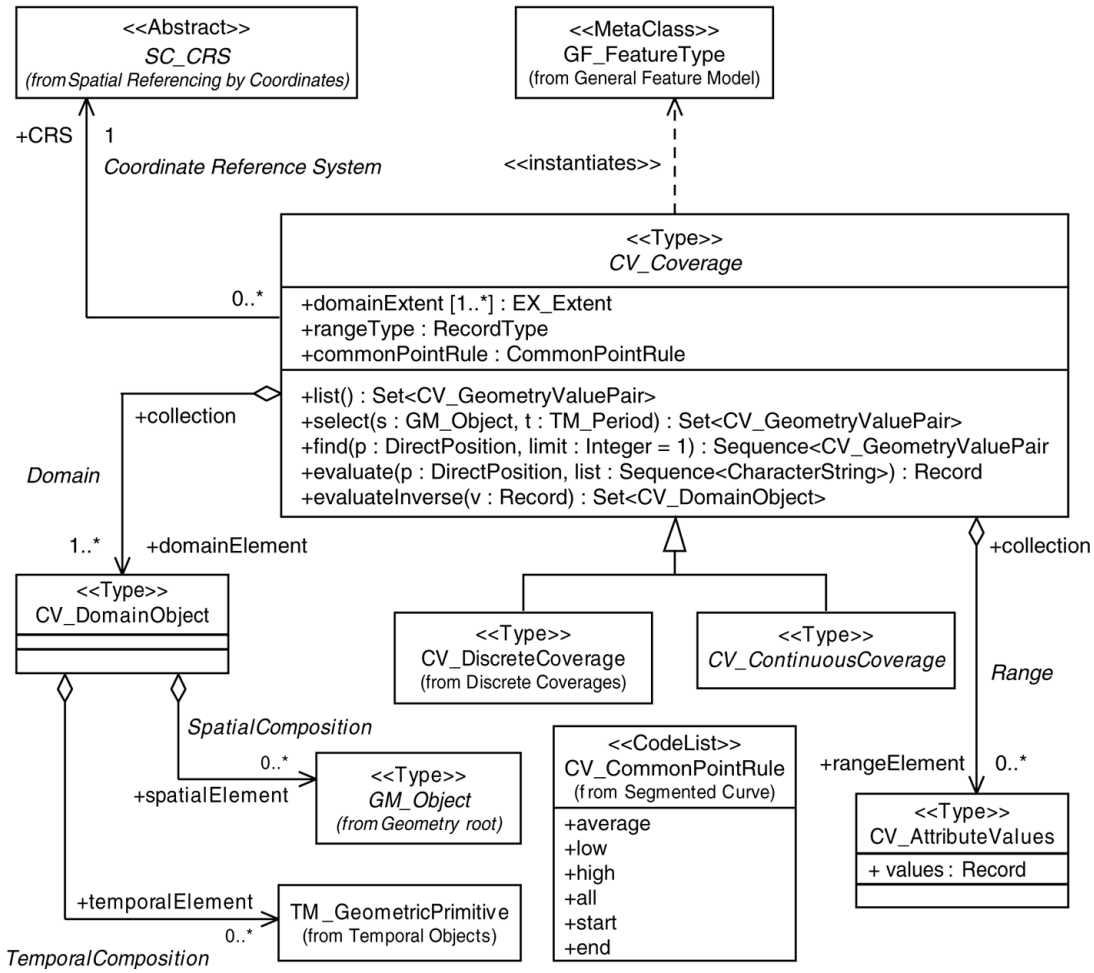


Figure 2 — CV_Coverage

5.3.9 evaluateInverse

The operation *evaluateInverse (v: Record): Set <CV_DomainObject>* shall accept a Record of feature attribute values as input and return a set of CV_DomainObjects. Normally, this will be the set of CV_DomainObjects in the Domain that are associated with values equal to those in the input Record. However, the operation may return other CV_DomainObjects derived from those in the domain, as specified by the application schema.

EXAMPLE The *evaluateInverse* operation could return a set of contours derived from the feature attribute values associated with the CV_GridPoints of a CV_GridCoverage.

5.3.10 Coordinate Reference System

The association *Coordinate Reference System* shall link the CV_Coverage to the coordinate reference system to which the objects in its domain are referenced. The class SC_CRS is specified in ISO 19111:2003. The multiplicity of the CRS role in the Coordinate Reference System association is one, so a coverage with the same range but with its domain defined in a different coordinate reference system is a different coverage.

5.3.11 Domain

The association *Domain* shall link the CV_Coverage to the set of CV_DomainObjects in the domain.

5.3.12 Range

The association *Range* shall link the CV_Coverage to the set of CV_AttributeValues in the range. The range of a CV_Coverage shall be a homogeneous collection of records. That is, the range shall have a constant dimension over the entire domain, and each field of the record shall provide a value of the same attribute type over the entire domain.

NOTE This International Standard does not specify how the Domain and Range associations are to be implemented. The relevant data may be generated in real time, it may be held in persistent local storage, or it may be electronically accessible from remote locations.

5.4 CV_DomainObject

5.4.1 General

CV_DomainObject represents an element of the domain of the CV_Coverage. It is an aggregation of objects that may include any combination of GM_Objects (ISO 19107:2003), TM_GeometricPrimitives (ISO 10108), or spatial or temporal objects defined in other standards, such as the CV_GridPoint defined in this International Standard.

5.4.2 SpatialComposition

The association *SpatialComposition* shall associate a CV_DomainObject to the set of GM_Objects of which it is composed.

5.4.3 TemporalComposition

The association *TemporalComposition* shall associate a CV_DomainObject to the set of TM_GeometricPrimitives of which it is composed.

5.5 CV_AttributeValues

5.5.1 General

CV_AttributeValues represents an element from the range of the CV_Coverage.

5.5.2 values

The attribute *values* is a Record containing one value for each attribute, as specified in CV_Coverage.rangeType (5.3.3).

EXAMPLES A coverage with a single (scalar) value (such as elevation). A coverage with a series (array/tensor) of values all defined in the same way (such as brightness values in different parts of the electromagnetic spectrum).

5.5.3 Range

The association *Range* shall link the set of CV_AttributeValues to the CV_Coverage that has the set as its range (5.3.12).

In the case of a discrete coverage, the multiplicity of CV_Coverage.rangeElement equals that of CV_Coverage.domainElement. In other words, there is one instance of CV_AttributeValues for each instance of CV_DomainObject. Usually, these are stored values that are accessed by the *evaluate* operation.

In the case of a continuous coverage, there is a transfinite number of instances of CV_AttributeValues for each CV_DomainObject. A few instances may be stored as input for the *evaluate* operation, but most are generated as needed by that operation.

5.6 CV_CommonPointRule

CV_CommonPointRule is a list of codes that identify methods for handling cases where the DirectPosition input to the *evaluate* operation falls within two or more of the geometric objects. The interpretation of these rules differs between discrete and continuous coverages. In the case of a discrete coverage, each CV_GeometryValuePair provides one value for each attribute. The rule is applied to the set of values associated with the set of CV_GeometryValuePairs that contain the DirectPosition. In the case of a continuous coverage, a value for each attribute shall be interpolated for each CV_ValueObject that contains the DirectPosition. The rule shall then be applied to the set of interpolated values for each attribute. The codes and their meanings are:

a) average – return the mean of the feature attribute values;

b) low – use the least of the feature attribute values;

c) high – use the greatest of the feature attribute values;

d) all – return all the feature attribute values that can be determined for the input DirectPosition;

e) start – use the *startValue* of the second CV_ValueSegment;

f) end – use the *endValue* of the first CV_ValueSegment.

NOTE The codes “start” and “end” apply only to segmented curve coverages.

5.7 CV_DiscreteCoverage

5.7.1 General

Figure 3 describes the principal subclasses of CV_Coverage. CV_DiscreteCoverage is the subclass that returns the same record of feature attribute values for any direct position within a single CV_DomainObject in its domain. Subclasses of CV_DiscreteCoverage are described in Clause 6.

5.7.2 locate

The operation *locate (p: DirectPosition): Set <CV_GeometryValuePair>* shall accept a DirectPosition as input and return the set of CV_GeometryValuePairs that include CV_DomainObjects containing the DirectPosition. It shall return a null value if the DirectPosition is not on any of the CV_DomainObjects within the domain of the CV_DiscreteCoverage.

5.7.3 evaluate

The operation *evaluate (p: DirectPosition, list: Sequence <CharacterString>): Set <Record>*, which is inherited from CV_Coverage, shall accept a DirectPosition as input, locate the CV_GeometryValuePairs that include the CV_DomainObjects that contain the DirectPosition, and return a set of records of feature attribute values. Normally, the input DirectPosition will fall within only one CV_GeometryValuePair, and the operation will return the record of feature attribute values associated with that CV_GeometryValuePair. If the DirectPosition falls on the boundary between two CV_GeometryPairs, or within two or more overlapping CV_GeometryValuePairs, the operation shall return a record of feature attribute values derived according to the value of the attribute *commonPointRule*. It shall return a null value if the DirectPosition is not on any of the CV_DomainObjects within the domain of the CV_DiscreteCoverage.

5.7.4 evaluateInverse

The operation *evaluateInverse (v: Record): Set <CV_DomainObject>*, which is inherited from CV_Coverage, shall accept a Record of feature attribute values as input, locate the CV_GeometryValuePairs for which *value* equals the input record, and return the set of CV_DomainObjects belonging to those CV_GeometryValuePairs. It shall return a null value if none of the CV_GeometryValuePairs associated with the CV_DiscreteCoverage has a *value* equal to the input Record.

5.7.5 CoverageFunction

The association *CoverageFunction* shall link the CV_DiscreteCoverage to the set of CV_GeometryValuePairs included in the coverage.

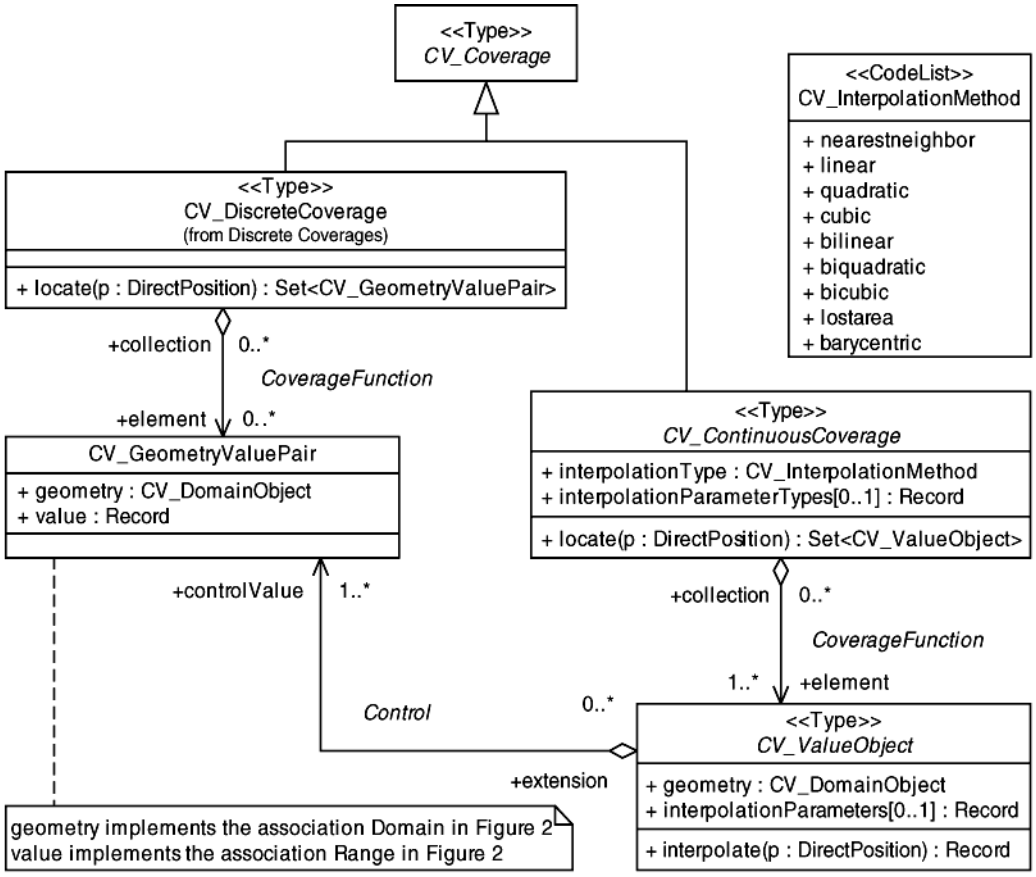


Figure 3 — CV_Coverage subclasses

5.8 CV_GeometryValuePair

5.8.1 General

The class CV_GeometryValuePair describes an element of a set that defines the relationships of a discrete coverage. Each member of this class consists of two parts: a domain object from the domain of the coverage to which it belongs and a record of feature attribute values from the range of the coverage to which it belongs. CV_GeometryValuePairs may be generated in the execution of an *evaluate* operation, and need not be persistent. CV_GeometryValuePair is subclassed (Clause 6) to restrict the pairing of a feature attribute value record to a specific subtype of domain object.

5.8.2 geometry

The attribute *geometry: CV_DomainObject* shall hold the CV_DomainObject that is a member of this CV_GeometryValuePair.

5.8.3 value

The attribute *value: Record* shall hold the record of feature attribute values associated with this CV_DomainObject.

5.8.4 CoverageFunction

The association *CoverageFunction* shall link this CV_GeometryValuePair with the CV_DiscreteCoverage of which it is an element.

5.8.5 Control

The association *Control* is empty in the case of a discrete coverage. It is described in the context of CV_ValueObject (5.10.5).

5.9 CV_ContinuousCoverage

5.9.1 General

CV_ContinuousCoverage is the subclass of CV_Coverage that returns a distinct record of feature attribute values for any direct position within its domain.

5.9.2 interpolationType

The attribute *interpolationType [0..1]: CV_InterpolationMethod* shall be a code that identifies the interpolation method that shall be used to derive a feature attribute value at any direct position within the CV_ValueObject. The attribute is optional – no value is needed for an analytical coverage (one that maps direct position to attribute value by using a mathematical function rather than by interpolation).

5.9.3 interpolationParameterTypes

Although many interpolation methods use only the values in the coverage range as input to the interpolation function, there are some methods that require additional parameters. The optional attribute *interpolationParameterTypes* specifies the types of parameters that are needed to support the interpolation method identified by *interpolationType*. The data type RecordType is specified in ISO/TS 19103. It is a dictionary of names and data types.

5.9.4 locate

The operation *locate (p: DirectPosition): Set <CV_ValueObject>* shall accept a DirectPosition as input and return the set of CV_ValueObjects that contains this DirectPosition. It shall return a null value if the DirectPosition is not in any of the CV_ValueObjects within the domain of the CV_DiscreteCoverage.

5.9.5 select

The operation *select* is inherited from CV_Coverage (5.3.6). In the case of CV_ContinuousCoverage, the CV_DomainObjects that shall be returned are those belonging to the CV_GeometryValuePairs associated with the CV_Value Objects of which the CV_ContinuousCoverage is composed.

5.9.6 evaluate

The operation *evaluate (p: DirectPosition, list:Sequence <CharacterString>): Record*, which is inherited from CV_Coverage, shall accept a DirectPosition as input and return a record of feature attribute values for that direct position. Most evaluation methods involve interpolation within or around a CV_ValueObject. Normally, the input DirectPosition will fall within only one CV_ValueObject, and the operation will return a record of feature attribute values interpolated within that CV_ValueObject. If the DirectPosition falls on the boundary between two CV_ValueObjects, or within two or more overlapping CV_ValueObjects, the operation shall return a record of feature attribute values derived according to the value of the attribute *commonPointRule*. It shall return a null value if the DirectPosition is not on any CV_ValueObject.

5.9.7 evaluateInverse

The operation *evaluateInverse (v: record): Set <CV_DomainObject>*, which is inherited from CV_Coverage, shall accept a Record of feature attribute values as input, locate the CV_GeometryValuePairs for which *value* equals the input record, and return the set of CV_DomainObjects belonging to those CV_GeometryValuePairs. Normally, the CV_DomainObjects that shall be returned are those belonging to the CV_GeometryValuePairs associated with the CV_Value Objects of which the CV_ContinuousCoverage is composed. However, the operation may return other CV_DomainObjects derived from those in the domain, as specified by the application schema. The operation shall return a null value if none of the CV_GeometryValuePairs associated with the CV_DiscreteCoverage has a *value* equal to the input Record.

EXAMPLE The *evaluateInverse* operation could return a set of contours derived from the feature attribute values associated with the CV_GridPoints of a CV_GridCoverage.

5.9.8 CoverageFunction

The association *CoverageFunction* shall link this CV_ContinuousCoverage to the set of CV_ValueObjects used to evaluate the coverage. This association is optional – an analytical coverage needs no CV_ValueObjects.

5.10 CV_ValueObject

5.10.1 General

CV_ValueObject provides a basis for interpolating feature attribute values within a CV_ContinuousCoverage. CV_ValueObjects may be generated in the execution of an *evaluate* operation, and need not be persistent.

5.10.2 geometry

The attribute *geometry: CV_DomainObject* is a CV_DomainObject constructed from the CV_DomainObjects of the CV_GeometryValuePairs that are linked to this CV_ValueObject by the association *Control*.

5.10.3 interpolationParameters

The optional attribute *interpolationParameters: Record* shall hold the values of the parameters required to execute the *interpolate* operation, as specified by the *interpolationParameterTypes* attribute of CV_ContinuousCoverage.

5.10.4 interpolate

The operation *interpolate (p: DirectPosition): Record* shall accept a DirectPosition as input and return the record of feature attribute values computed for that DirectPosition.

5.10.5 Control

The association *Control* shall link this CV_ValueObject to the set of CV_GeometryValuePairs that provide the basis for constructing the CV_ValueObject and for evaluating the CV_ContinuousCoverage at DirectPositions within this CV_ValueObject.

5.10.6 CoverageFunction

The association *CoverageFunction* shall link this CV_ValueObject to the CV_ContinuousCoverage of which it is an element.

5.11 CV_InterpolationMethod

CV_InterpolationMethod is a list of codes that identify interpolation methods that may be used for evaluating continuous coverages. See Annex C for descriptions of specific interpolation methods.

5.12 Subclasses of CV_ContinuousCoverage

This International Standard specifies schemas for five subclasses of CV_ContinuousCoverage (Figure 4). CV_ThiessenPolygonCoverage is specified in Clause 7, CV_ContinuousQuadrilateralGridCoverage is specified in Clause 8, CV_HexagonalGridCoverage is specified in Clause 9, CV_TINCoverage is specified in Clause 10, and CV_SegmentedCurveCoverage is specified in Clause 11.

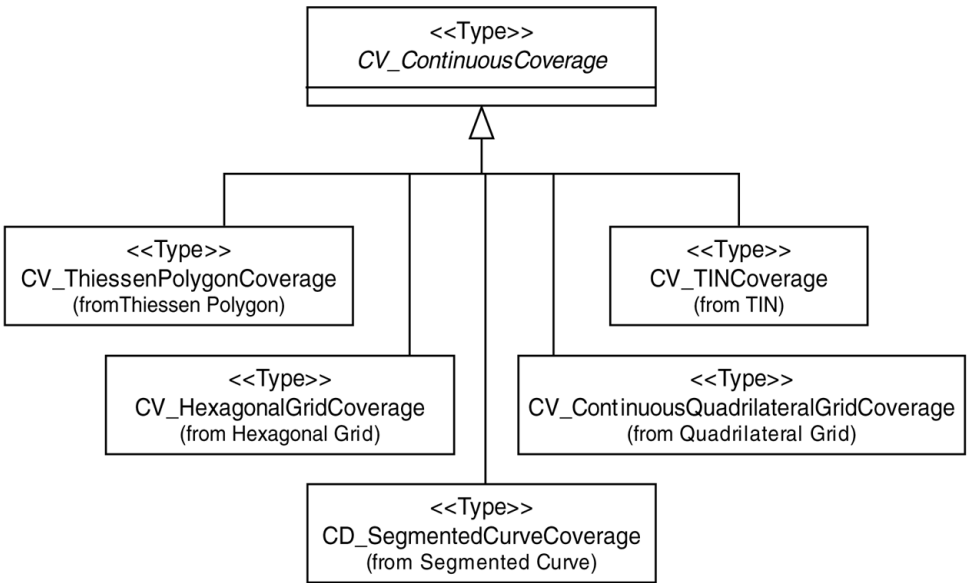


Figure 4 — Continuous coverages

6 Discrete coverages

6.1 Discrete coverage types

The domain of a CV_DiscreteCoverage consists of a collection of geometric objects. CV_DiscreteCoverages are subclassed on the basis of the type of geometric object in the spatial domain (Figure 5). Each subclass of CV_DiscreteCoverage is associated with a specific subclass of CV_GeometryValuePair. The subclasses of both classes inherit the attributes and operations specified for the parent classes, and the association between the parent classes, but with the restrictions described below.

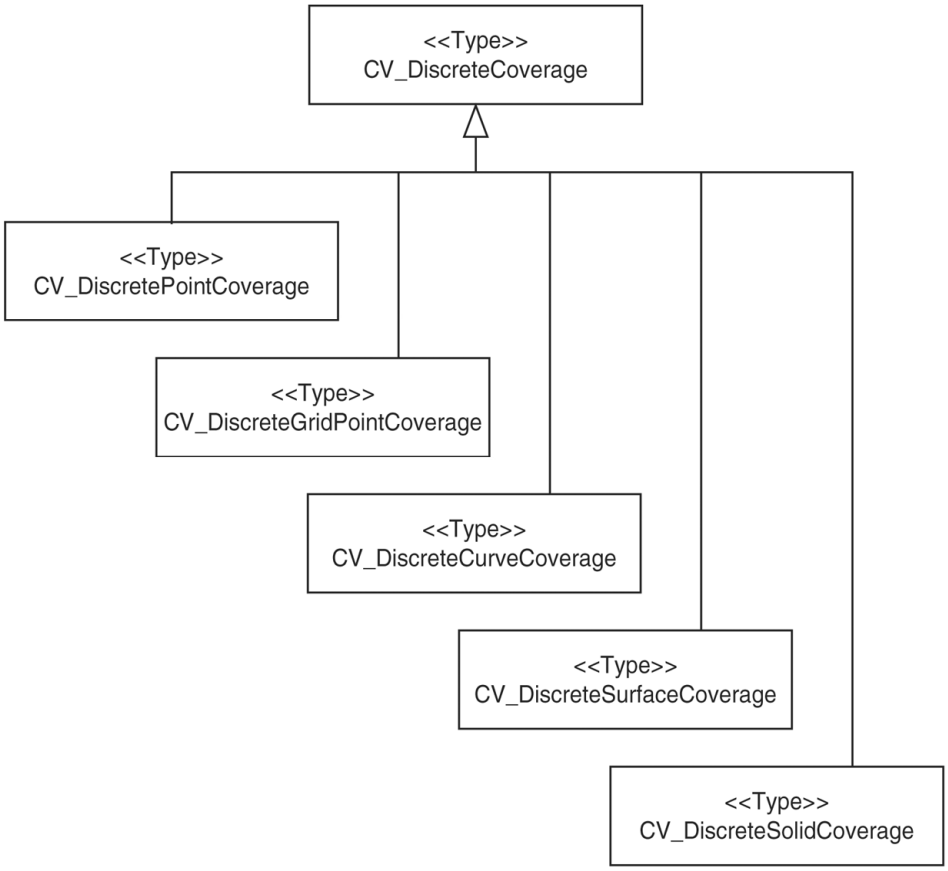


Figure 5 — Discrete coverage types

Because the superclass is not abstract, an instance of the superclass may consist of mixed types of CV_GeometryValuePair.

6.2 CV_DiscretePointCoverage

6.2.1 General

A discrete point coverage is characterized by a finite domain consisting of points. Generally, the domain is a set of irregularly distributed points. However, the principal use of discrete point coverages is to provide a basis for continuous coverage functions, where the evaluation of the continuous coverage function is accomplished by interpolation between the points of the discrete point coverage. Most interpolation algorithms depend upon a structured pattern of spatial relationships between the points. This requires either that the points in the spatial domain of the discrete point coverage be arranged in a regular way, or that the spatial domain of the continuous coverage be partitioned in a regular way in relation to the points of the discrete point coverage. Grid coverages (Clauses 8 and 9) employ the first method; Thiessen polygon (Clause 7) and TIN (Clause 10) coverages employ the second.

EXAMPLE A set of hydrographic soundings is a discrete point coverage.

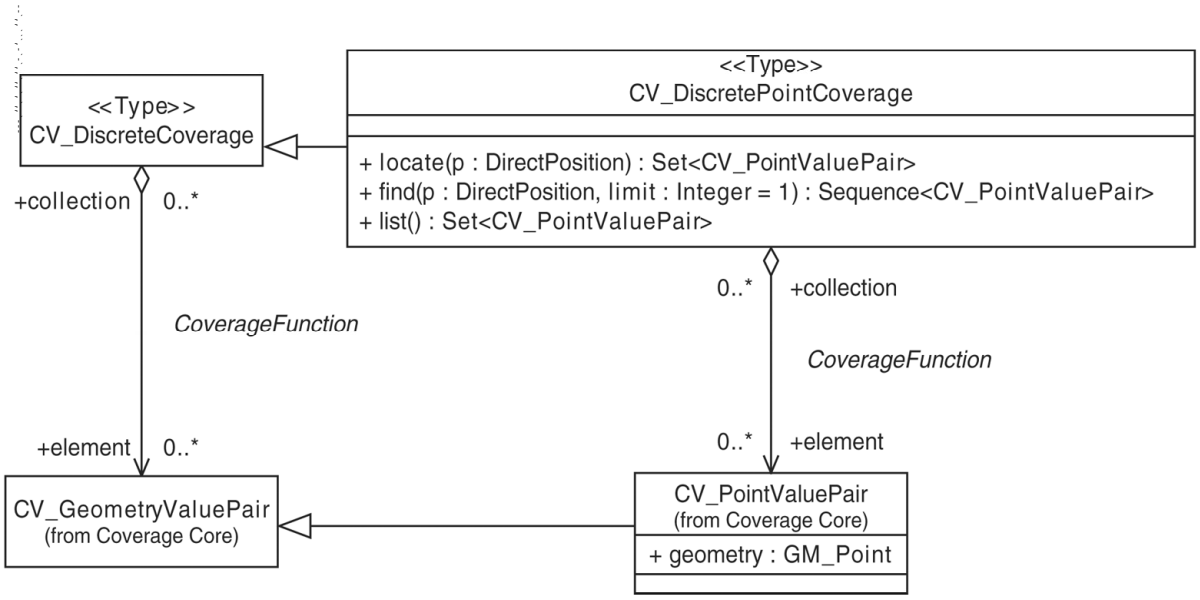


Figure 6 — CV_DiscretePointCoverage

6.2.2 Inherited associations and operations

CV_DiscretePointCoverage (Figure 6) inherits the association *CoverageFunction* and the operations *locate*, *find*, and *list* from CV_DiscreteCoverage (5.7), with the restriction that the associated CV_GeometryValuePairs and those returned by the operations shall be limited to CV_PointValuePairs.

6.3 CV_PointValuePair

CV_PointValuePair is the subtype of CV_GeometryValuePair that has a GM_Point as the value of its geometry attribute.

6.4 CV_DiscreteGridPointCoverage

6.4.1 General

The domain of a CV_DiscreteGridPointCoverage (Figure 7) is a set of CV_GridPoints (8.5) that are associated with records of feature attribute values through a CV_GridValuesMatrix (8.14).

6.4.2 Inherited associations and operations

CV_DiscreteGridPointCoverage (Figure 7) inherits the association *CoverageFunction* and the operations *locate*, *find*, and *list*, from CV_DiscreteCoverage, with the restriction that the associated CV_GeometryValuePairs and those returned by the operations shall be limited to CV_GridPointValuePairs. The association *CoverageFunction* is shown as derived in this case because the *element*s may be generated from the CV_GridValuesMatrix through the association *PointFunction*. The inherited operations *evaluate* and *evaluateInverse* use CV_GridValuesMatrix to assign values to the CV_GeometryValuePairs.

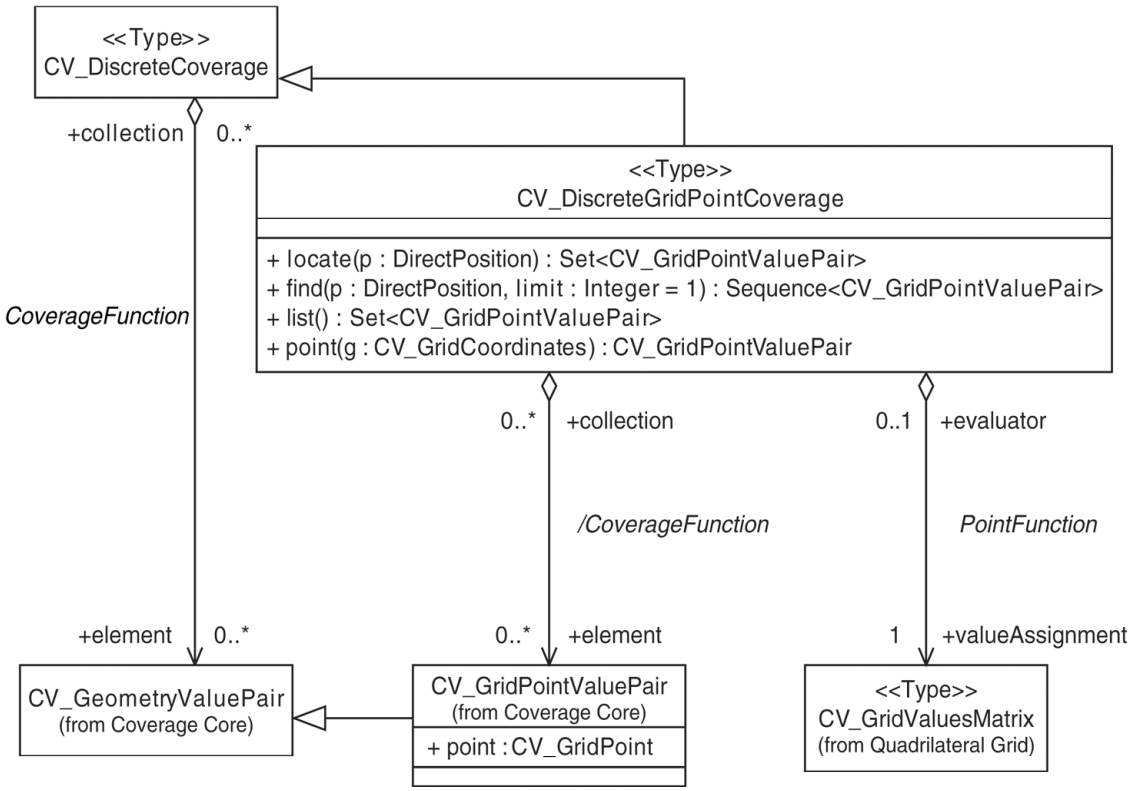


Figure 7 — CV_DiscreteGridPointCoverage

6.4.3 point

The operation *point (g: CV_GridCoordinates): CV_GridPointValuePair* shall accept a grid coordinate as input and use data from the associated CV_GridValuesMatrix to construct and return the CV_GridPointValuePair associated with that grid position.

6.4.4 PointFunction

The association *PointFunction* shall link the CV_DiscreteGridPointCoverage to the CV_GridValuesMatrix for which it is an evaluator.

6.5 CV_GridPointValuePair

CV_GridPointValuePair is the subtype of CV_GeometryValuePair that has a GM_GridPoint as the value of its geometry attribute.

6.6 CV_DiscreteCurveCoverage

6.6.1 General

A discrete curve coverage is characterized by a finite spatial domain consisting of curves. Often the curves represent features such as roads, railroads or streams. They may be elements of a network.

EXAMPLE A coverage that assigns a route number, a name, a pavement width and a pavement material type to each segment of a road system.

6.6.2 Inherited operations and associations

CV_DiscreteCurveCoverage (Figure 8) inherits the association *CoverageFunction* and the operations *locate*, *find*, *list, evaluate* and *evaluateInverse* from CV_DiscreteCoverage, with the restriction that the associated CV_GeometryValuePairs and those returned by the operations shall be limited to CV_CurveValuePairs.

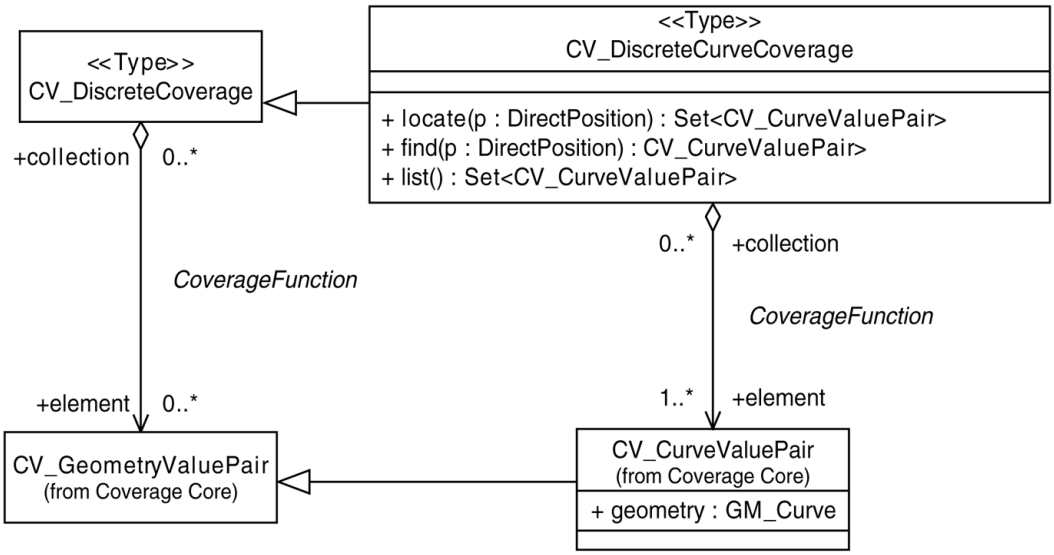


Figure 8 — CV_DiscreteCurveCoverage

6.7 CV_CurveValuePair

CV_CurveValuePair is the subtype of CV_GeometryValuePair that has a GM_Curve as the value of its geometry attribute.

6.8 CV_DiscreteSurfaceCoverage

6.8.1 General

A discrete surface coverage is a coverage whose domain consists of a collection of surfaces. In most cases, the surfaces that constitute the domain of a coverage are mutually exclusive and exhaustively partition the extent of the coverage. Surfaces or their boundaries may be of any shape. The boundaries of component surfaces often correspond to natural phenomena and are highly irregular.

EXAMPLE A coverage that represents soil types typically has a spatial domain composed of surfaces with irregular boundaries.

Any set of polygons can be used as a spatial domain for a discrete surface coverage. Spatial domains composed of congruent polygons are very common. Often, these domains are composed of congruent rectangles or regular hexagons. The geometry of such a tessellation may be described in terms of a quadrilateral grid (8.2) or a hexagonal grid (9.1). The spatial domain of a discrete surface coverage may also consist of the triangles that compose a TIN (10.7), or the polygons of a Thiessen polygon network (7.1).

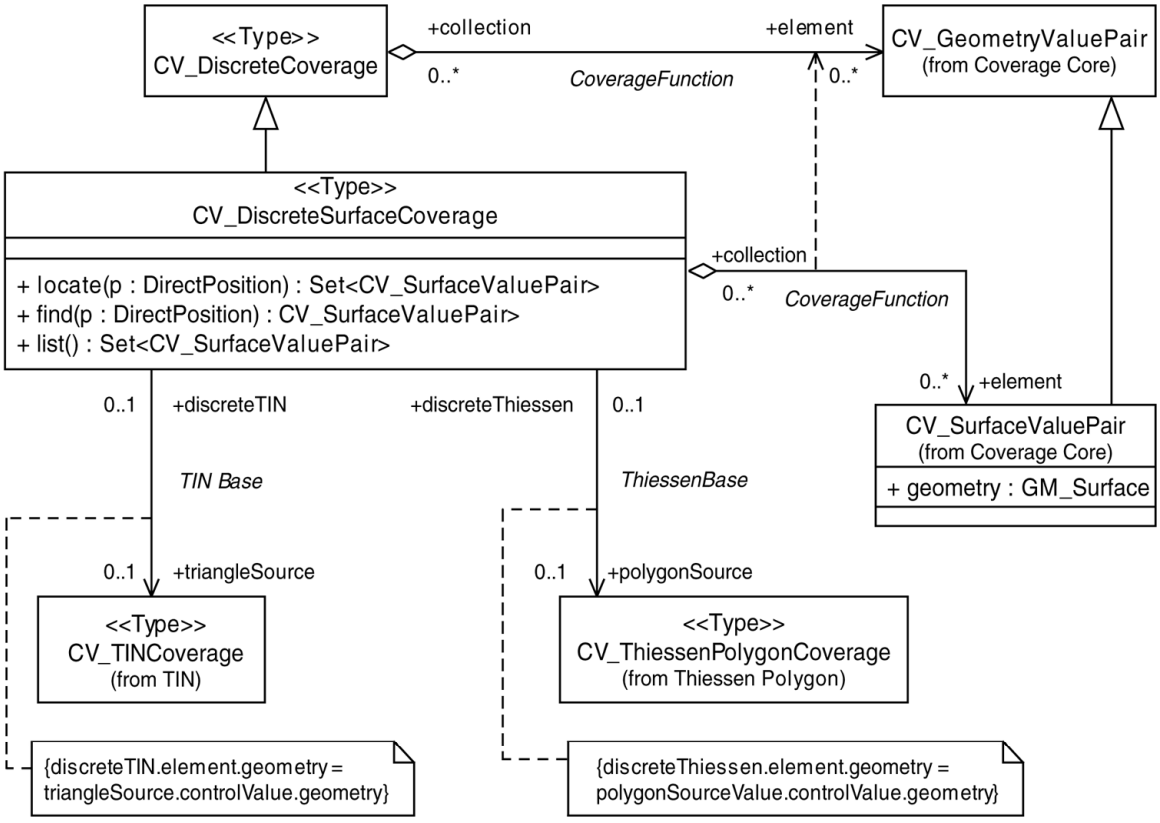


Figure 9 — CV_DiscreteSurfaceCoverage

6.8.2 Inherited operations and associations

CV_DiscreteSurfaceCoverage (Figure 9) inherits the association *CoverageFunction* and the operations *locate*, *find*, *list*, *evaluate*, and *evaluateInverse* from CV_DiscreteCoverage, with the restriction that the associated CV_GeometryValuePairs and those returned by the operations shall be limited to CV_SurfaceValuePairs.

6.8.3 TINBase

The association *TINBase* may be used to link a CV_DiscreteSurfaceCoverage to a CV_TINCoverage (10.2). The constraint

discreteTIN.element.geometry = triangleSource.controlValue.geometry

requires that the spatial domain of the CV_DiscreteSurfaceCoverage be composed of the triangles belonging to the CV_TINCoverage.

6.8.4 ThiessenBase

The association *ThiessenBase* may be used to link a CV_DiscreteSurfaceCoverage to a CV_ThiessenPolygonCoverage (7.2). The constraint

discreteThiessen.element.geometry = polygonSource.controlValue.geometry

requires that the spatial domain of the CV_DiscreteSurfaceCoverage be composed of the polygons belonging to the CV_ThiessenPolygonCoverage.

6.9 CV_SurfaceValuePair

CV_SurfaceValuePair is the subtype of CV_GeometryValuePair that has a GM_Surface as the value of its geometry attribute.

6.10 CV_DiscreteSolidCoverage

6.10.1 General

A discrete solid coverage is a coverage whose domain consists of a collection of solids. Solids or their boundaries may be of any shape. Generally, the solids that constitute the domain of a coverage are mutually exclusive and exhaustively partition the extent of the coverage, but this is not required.

EXAMPLE Buildings in an urban area could be represented as a set of unconnected GM_Solids each with attributes such as building name, address, floor space and number of occupants.

As in the case of surfaces (5.8), the spatial domain of a discrete solid coverage may be a regular or semiregular tessellation of the extent of the coverage. The tessellation can be defined in terms of a three-dimensional grid, where the set of grid cells is the spatial domain of the coverage.

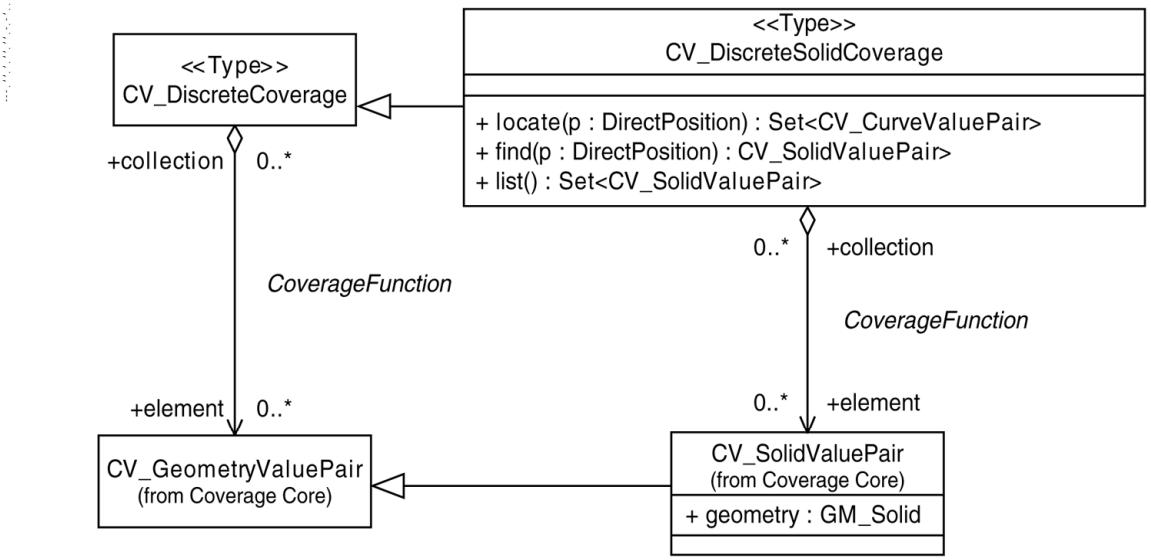


Figure 10 — CV_DiscreteSolidCoverage

6.10.2 Inherited operations and associations

CV_DiscreteSolidCoverage (Figure 10) inherits the association *CoverageFunction* and the operations *locate*, *find*, *list*, *evaluate* and *evaluateInverse* from CV_DiscreteCoverage, with the restriction that the associated CV_GeometryValuePairs and those returned by the operations shall be limited to CV_SolidValuePairs.

6.11 CV_SolidValuePair

CV_SolidValuePair is the subtype of CV_GeometryValuePair that has a GM_Solid as the value of its geometry attribute.

7 Thiessen polygon coverage

7.1 Thiessen polygon networks

A finite collection of points on a plane determines a partition of the plane into a collection of polygons equal in number to the collection of points. A Thiessen polygon is generated from one of a defining set of points by forming the set of direct positions that are closer to that point than to any other point in the defining set. The specific point is called the centre of the resulting polygon. The boundaries between neighbouring polygons are the perpendicular bisectors of the lines between their respective centres. Each polygon shares each of its edges with exactly one other polygon. Each polygon contains exactly one point from the defining set. Thiessen polygons are also known as Voronoi Diagrams or Proximal Sets.

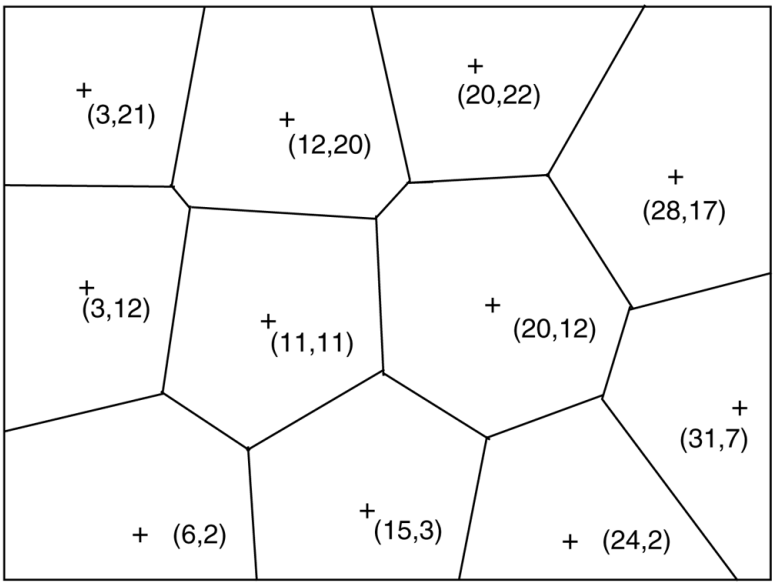


Figure 11 — An example of a Thiessen polygon network with (x,y) coordinates

A Thiessen polygon network (Figure 11) is a tessellation of a two-dimensional space using Thiessen Polygons. A Thiessen polygon network provides a structure that supports interpolation of feature attribute values from the polygon centres to direct positions within the polygons.

EXAMPLE Figure 11 shows a collection of points with their (x,y) coordinates, the perpendicular bisectors of the lines that would be drawn between them, and the resultant polygons.

7.2 CV_ThiessenPolygonCoverage

7.2.1 General

A CV_ThiessenPolygonCoverage (Figure 12) evaluates a coverage at direct positions within a Thiessen polygon network constructed from a set of discrete point value pairs. Evaluation is based on interpolation between the centres of the CV_ThiessenValuePolygons surrounding the input position.

7.2.2 clipArea

The attribute *clipArea: GM_Surface* shall describe the extent of the CV_ThiessenPolygonNetwork. Its boundary determines the boundaries of the outermost polygons in the network, which would otherwise be unbounded.

7.2.3 interpolationType

The inherited attribute *interpolationType: CV_InterpolationMethod = "lost area"* shall identify the interpolation method to be used in evaluating the coverage. The most common interpolation methods are "lost area" (C.9) and "nearest neighbour" (C.2). Lost area interpolation can return a different Record of feature attribute values for each direct position within a CV_ThiessenValuePolygon. On the other hand, nearest neighbour interpolation will return for any direct position within a CV_ThiessenValuePolygon the Record associated with the CV_PointValuePair at the centre of the CV_ThiessenPolygon. In other words, a CV_ThiessenPolygonCoverage that uses nearest neighbour interpolation acts like a discrete surface coverage.

7.2.4 locate

The operation *locate (p: DirectPosition): CV_ThiessenValuePolygon* is inherited from CV_ContinuousCoverage with the restriction that it shall return a CV_ThiesenValuePolygon. It shall accept a DirectPosition as input and return the CV_ThiessenValuePolygon that contains that DirectPosition.

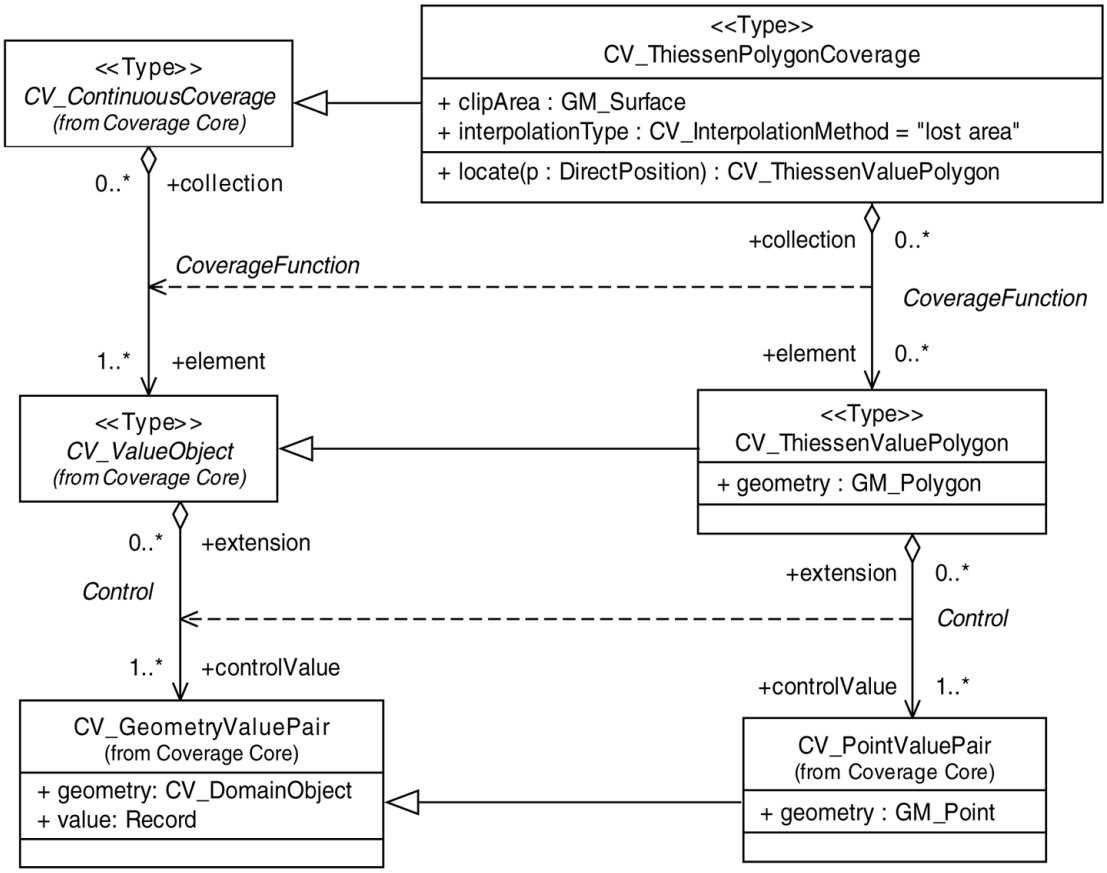


Figure 12 — CV_ThiessenPolygonCoverage

7.2.5 evaluate

The operation *evaluate (p: DirectPosition, list: Sequence <CharacterString>): Record* is inherited from CV_Coverage. Evaluation of a CV_ThiessenPolygonCoverage involves two steps. The first is to locate the CV_ThiessenValuePolygon that contains the input DirectPosition; the second is to interpolate the feature attribute values at the DirectPosition from the CV_PointValuePairs at the centres of the surrounding CV_ThiessenValuePolygons.

7.2.6 CoverageFunction

The association *CoverageFunction* shall link this CV_ThiessenPolygonCoverage to the CV_ThiessenValuePolygons of which it is composed.

7.3 CV_ThiessenValuePolygon

7.3.1 General

CV_ThiessenValuePolygon is a subclass of CV_ValueObject. Individual CV_ThiessenValuePolygons may be generated during the evaluation of a CV_ThiessenPolygonCoverage, and need not be persistent.

7.3.2 geometry

The attribute *geometry:GM_Polygon* shall hold the geometry of the Thiessen polygon centred on the CV_PointValuePair identified by the association *Control.*

7.3.3 Control

The association *Control* shall link this CV_ThiessenValuePolygon to the CV_PointValuePair at its centre.

8 Quadrilateral grid coverages

8.1 General

Grid coverages employ a systematic tessellation of the domain. The principal advantage of such tessellations is that they support a sequential enumeration of the elements of the domain, which makes data storage and access more efficient. The tessellation may represent how the data were acquired or how they were computed in a model. The domain of a grid coverage is a set of grid points, including their convex hull in the case of a continuous grid coverage.

8.2 Quadrilateral grid geometry

8.2.1 General

A grid is a network composed of two or more sets of curves in which the members of each set intersect the members of the other sets in a systematic way. The curves are called grid lines; the points at which they intersect are grid points, and the interstices between the grid lines are grid cells.

The most common case is the one in which the curves are straight lines, and there is one set of grid lines for each dimension of the grid space. In this case, the grid cells are parallelograms or parallelepipeds. In its own coordinate system, such a grid 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 (Figure 13). It has a set of axes equal in number to the dimension of the grid. It has one set of grid lines parallel to each axis. The size of the grid is described by a sequence of integers, in which each integer is a count of the number of lines parallel to one of the axes. There are grid points at all grid line intersections. The axes of the grid provide a basis for defining grid coordinates, which are measured along the axes away from their origin, which is distinguished by having coordinate values of 0. Grid coordinates of grid points are integer numbers. The axes need to be identified to support sequencing rules for associating feature attribute value records to the grid points.

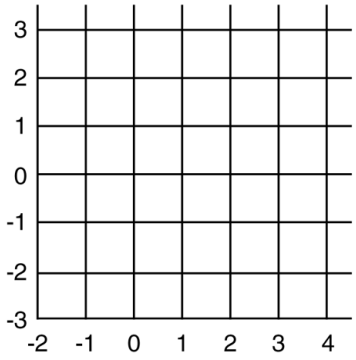


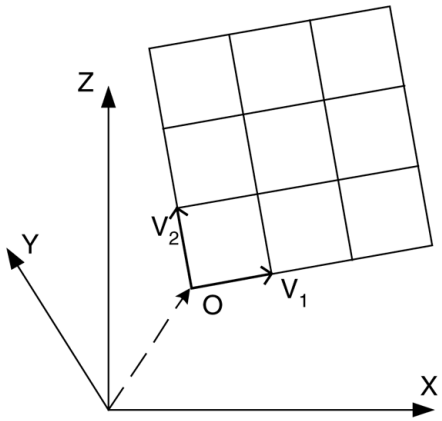
Figure 13 — Example — A 7 × 7 two-dimensional orthogonal grid

NOTE The dimensions (axes) of a 2-dimensional grid are often called row and column.

A grid may be defined in terms of an external coordinate reference system. This requires additional information about the location of the grid's origin within the external coordinate reference system, the orientation of the grid axes, and a measure of the spacing between the grid lines. If the spacing is uniform, then there is an affine relationship between the grid and external coordinate system, and the grid (Figure 14) is called a rectified grid. If, in addition, the external coordinate reference system is related to the earth by a datum, the grid is a georectified grid. The grid lines of a rectified grid need not meet at right angles; the spacing between the grid lines is constant along each axis, but need not be the same on every axis. The essential point is that the transformation of grid coordinates to coordinates of the external coordinate reference system is an affine transformation.

NOTE 1 The word rectified implies a transformation from an image space to another coordinate reference system. However, grids of this form are often defined initially in an earth-based coordinate system and used as a basis for collecting data from sources other than imagery.

NOTE 2 The internal grid coordinate system is an instance of an engineering coordinate reference system as specified by ISO 19111:2003. Its datum is a set of one or more ground control points.



Key

X, Y, Z axes to determine 3-space

V_1, V_2 offset vectors

0 grid origin

Figure 14 — Geometry of a rectified grid

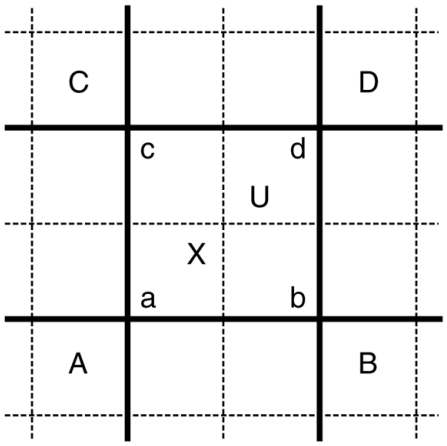
EXAMPLE Figure 14 shows a two-dimensional grid in the 3-space determined by the axes X, Y, and Z. The grid origin is at O. There are two offset vectors labelled V_1 and V_2 which specify the orientation of the grid axes and the spacing between the grid lines. The coordinates of the grid points are of the form: $O + aV_1 + bV_2$.

When the relationship between a grid and an external coordinate reference system is not adequate to specify it in terms of an origin, an orientation and spacing in that coordinate reference system, it may still be possible to transform the grid coordinates into coordinates in the coordinate reference system. This transformation need not be in analytic form; it may be a table, relating the grid points to coordinates in the external coordinate reference system. Such a grid is classified as a referenceable grid. If the external coordinate reference system is related to the earth by a datum, the grid is a georeferenceable grid. A referenceable grid is associated with information that allows the location of all points in the grid to be determined in the coordinate reference system, but the location of the points is not directly available from the grid coordinates, as opposed to a rectified grid where the location of the points in the coordinate reference system is derivable from the properties of the grid itself. The transformation produced by the information associated with a referenceable grid will produce a grid as seen in the coordinate reference system, but the grid lines of that grid need not be straight or orthogonal, and the grid cells may be of different shapes and sizes.

8.2.2 Cell structures

The term “grid cell” refers to two concepts: one important from the perspective of data collection and portrayal, the other important from the perspective of grid coverage evaluation. The ambiguity of this term is a common cause of positioning error in evaluating or portraying grid coverages.

The feature attribute values associated with a grid point represent characteristics of the real world measured or observed within a small space surrounding a sample point represented by the grid point. The grid lines connecting these points form a set of grid cells. A common simplifying assumption is that the sample space is equally divided among the sample points, so that the sample spaces are represented by a second set of cells congruent to the first, but offset so that each has a grid point at its centre. Evaluation of a grid coverage is based on interpolation between grid points, i.e. within a grid cell bounded by the grid lines that connect the grid points that represent the sample points.



Key

a, b, c, d	grid points
A, B, C, D	cells (bounded by dotted lines)
U	grid cell (bounded by solid lines)
X	direct position within the grid cell

Figure 15 — Grid cell structures

EXAMPLE In Figure 15, the intersections of the solid lines represent a set of grid points (a, b, c, d) that correspond to a set of sample points. The dotted lines bound the cells (A, B, C, D) that represent the sample spaces associated with these grid points. In this example, the sample spaces are offset grid cells, although they need not be. Evaluation at any direct position X within the grid cell U (bounded by the solid lines) will be based on interpolation from a, b, c and d (and possibly involve additional grid points outside the cell).

In this International Standard, the term grid cell refers to the cell bounded by the grid lines that connect the grid points. The term sample space refers to the observed or measured space surrounding a sample point. The term footprint refers to a representation of a sample space in the context of some coordinate reference system.

In dealing with gridded data, e.g. for processing or portrayal, it is often assumed that the size and shape of the sample spaces are a simple function of the spatial distribution of the sample points, and that the grid cells and the sample cells are congruent.

In fact, the size and shape of the sample space are determined by the method used to measure or calculate the attribute value. In the simplest case, the sample space is the sample point. It is often a disc, a sphere, or a hypersphere surrounding the sample point. In the case of sensed data, the size and shape of the sample space is also a function of the sensor model and its position relative to the sample point, and may be quite complex. Adjacent sample spaces may be coterminous or they may overlap or underlap.

In addition to affecting the size and shape of the sample space, the measurement technique affects the applicability of the observed or measured value to the sample space. It is often assumed that the recorded value represents the mean value for the sample space. In fact, elements of the sample space may not contribute uniformly to the result, so that it is better conceived as a weighted average where the weighting is a function of position within the sample space. Interpolation methods may be designed specifically to deal with characteristics of the sample space.

Transformation (e.g. rectification) between grid coordinates and an external coordinate reference system may distort the representation of the sample space in a way that causes interpolation errors.

8.3 CV_Grid

8.3.1 General

The class CV_Grid (Figure 16) contains the geometric characteristics of a quadrilateral grid.

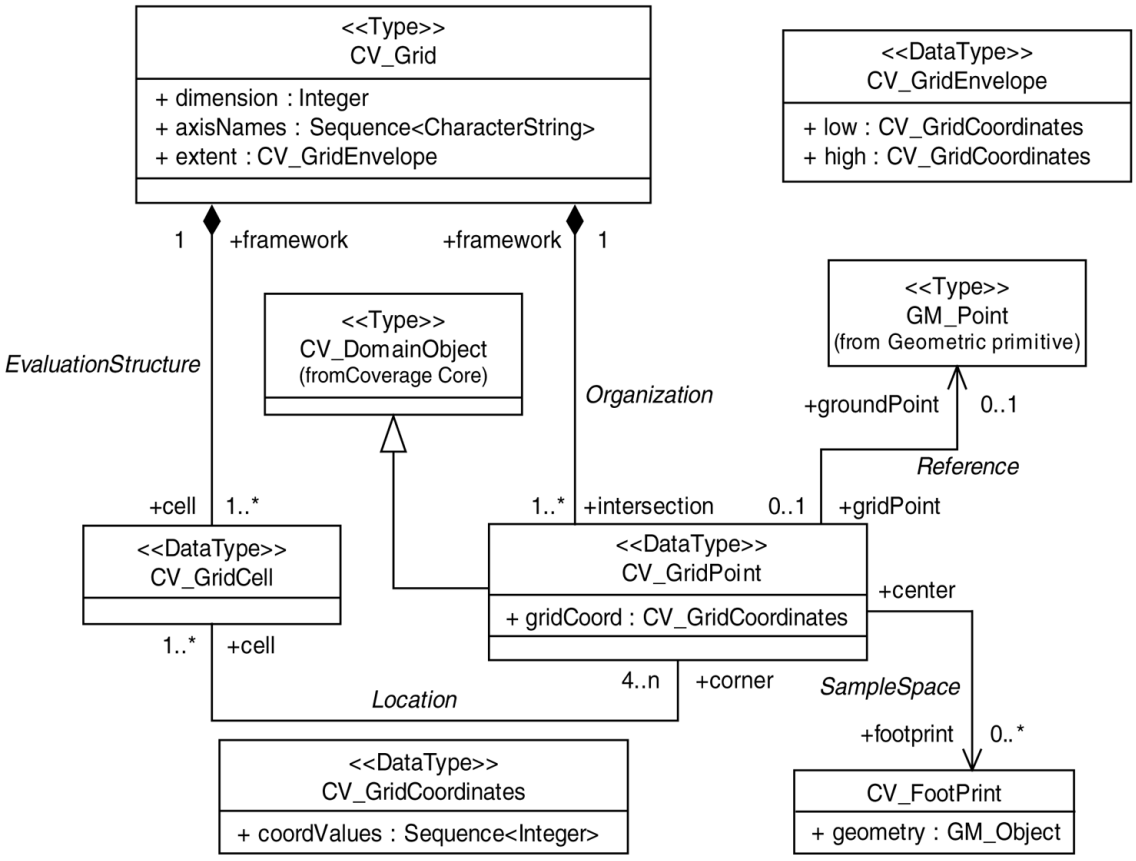


Figure 16 — CV_Grid

8.3.2 dimension

The attribute *dimension: Integer* shall identify the dimensionality of the grid.

8.3.3 axisNames

The attribute *axisNames: Sequence <CharacterString>* shall list the names of the grid axes.

8.3.4 extent

The optional attribute *extent: CV_GridEnvelope* shall specify the limits of a section of the grid.

8.3.5 Organization

The association *Organization* shall link the CV_Grid to the set of CV_GridPoints that are located at the intersections of the grid lines.

8.3.6 EvaluationStructure

The association *EvaluationStructure* shall link the CV_Grid to the set of CV_GridCells delineated by the grid lines.

8.3.7 Subclasses

CV_Grid has three subclasses (Figure 17), which lie in two partitions. The *Positioning* partition includes CV_RectifiedGrid (7.9) and CV_ReferenceableGrid (8.10), which contain information that relates the grid coordinates to an external coordinate reference system. The *Valuation* partition includes CV_GridValuesMatrix (8.14), which contains information for assigning values from the range to each of the grid points.

CV_Grid is not an abstract class: an instance of CV_Grid need not be an instance of any of its subclasses. The partitions indicate that an instance of the subclass CV_GridValuesMatrix may be, at the same time, an instance of either the subclass CV_RectifiedGrid or of the subclass CV_ReferenceableGrid.

8.4 CV_GridEnvelope

8.4.1 General

CV_GridEnvelope (Figure 16) is a data type that provides the grid coordinate values for the diametrically opposed corners of the CV_Grid. It has two attributes, low and high.

8.4.1.1 low

The attribute *low: CV_GridCoordinate* shall be the minimal coordinate values for all grid points within the CV_GridValuesMatrix.

8.4.1.2 high

The attribute *high: CV_GridCoordinate* shall be the maximal coordinate values for all grid points within the CV_GridValuesMatrix.

8.5 CV_GridPoint

8.5.1 General

CV_GridPoint is the class that represents the intersections of the grid lines.

8.5.2 gridCoord

The attribute *gridCoord: CV_GridCoordinate* holds the set of grid coordinates that specifies the location of the CV_GridPoint within the CV_Grid.

8.5.3 Organization

The association *Organization* shall link the CV_GridPoint to the CV_Grid of which it is an element.

8.5.4 Location

The association *Location* shall link the CV_GridPoint to the set of CV_GridCells for which it is a corner. The multiplicity at the CV_GridPoint end of the association has no upper bound, to allow for grids of any dimension. In a quadrilateral grid, the multiplicity of *corner* equals 2^d, where ^{d} is the value of *CV_Grid.dimension*.

8.5.5 Reference

The association *Reference* may link the CV_GridPoint to the GM_Point that is its representation in an external coordinate reference system.

8.5.6 SampleSpace

The association *SampleSpace* may link the CV_GridPoint to the CV_Footprint that represents the sample space in an external coordinate reference system associated with the CV_GridPoint. The multiplicity of the association SampleSpace allows for multiple external coordinate reference systems for CV_Footprint.

8.6 CV_GridCoordinate

8.6.1 General

CV_GridCoordinate is a data type for holding the grid coordinates of a CV_GridPoint.

8.6.2 coordValues

The attribute *coordValues: Sequence <Integer>* shall hold one integer value for each dimension of the grid. The ordering of these coordinate values shall be the same as that of the elements of *CV_Grid.axisNames*. The value of a single coordinate shall be the number of offsets from the origin of the grid in the direction of a specific axis.

8.7 CV_GridCell

8.7.1 General

A CV_GridCell is delineated by the grid lines of CV_Grid. Its corners are associated with the CV_GridPoints at the intersections of the grid lines that bound it.

8.7.2 Location

The association *Location* shall link the CV_GridCell to the set of CV_GridPoints at its corners.

8.7.3 EvaluationStructure

The association *EvaluationStructure* shall link the CV_GridCell to the CV_Grid of which it is a component.

8.8 CV_Footprint

8.8.1 General

A CV_Footprint is the sample space of a grid in an external coordinate reference system.

8.8.2 geometry

The attribute *geometry: GM_Object* shall describe the geometry of the CV_Footprint within the coordinate reference system identified by GM_Object.CRS (ISO 19107:2003).

8.8.3 SampleSpace

The association *SampleSpace* shall link the CV_Footprint to the CV_GridPoint to which it corresponds.

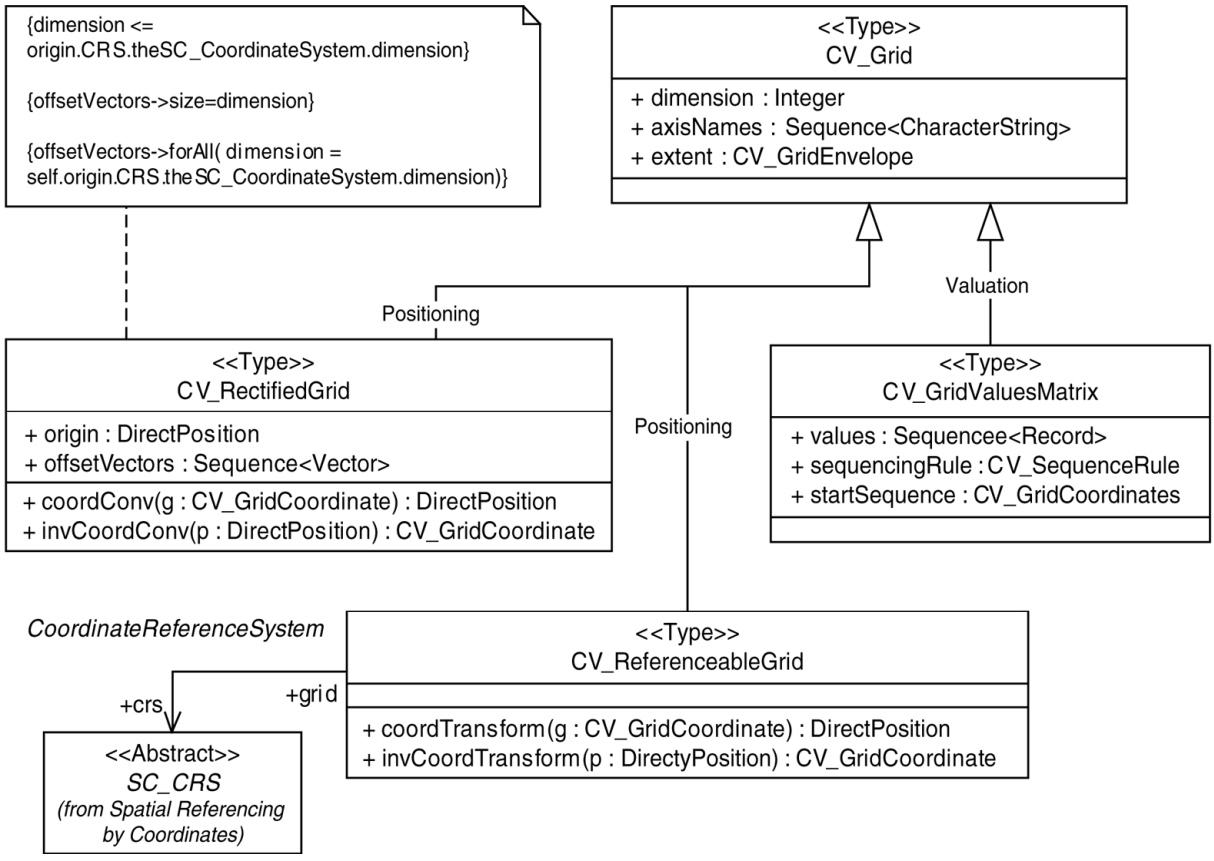


Figure 17 — CV_Grid subclasses

8.9 CV_RectifiedGrid

8.9.1 General

A rectified grid shall be defined by an origin in an external coordinate reference system, and a set of offset vectors that specify the direction and distance between the grid lines within that external coordinate reference system (Figure 14).

The class CV_RectifiedGrid (Figure 17) contains the additional geometric characteristics of a rectified grid.

8.9.2 origin

The attribute *origin: DirectPosition* is a direct position that shall locate the origin of the rectified grid in an external coordinate reference system. That coordinate reference system is identified through the association *Coordinate Reference System* (specified in ISO 19107:2003) between DirectPosition and the class SC_CRS specified in ISO 19111:2003.

8.9.3 offsetVectors

The attribute *offsetVectors: Sequence <Vector>* shall be a sequence of offset vectors that determine the grid spacing in each direction. The vectors are defined in terms of the external coordinate reference system. The data type Vector is specified in ISO/TS 19103.

8.9.4 coordConv

The operation *coordConv (g: CV_GridCoordinate): DirectPosition* shall accept a CV_GridCoordinate as input and return a DirectPosition. The operation uses the values of the attributes *origin* and *offsetVectors* in an affine transformation. It is a coordinate conversion operation as defined by ISO 19111:2003.

8.9.5 invCoordConv

The operation *invCoordConv (p: DirectPosition): CV_GridCoordinate* shall accept a DirectPosition as input and return the CV_GridCoordinate of the nearest CV_GridPoint. The operation uses the values of the attributes *origin* and *offsetVectors* in an affine transformation. It is a coordinate conversion operation as defined by ISO 19111:2003.

8.9.6 Constraints

a) {dimension <= origin.CRS.theSC_CoordinateSystem.dimension} The *dimension* of the grid shall be less than or equal to the dimension of the coordinate reference system identified through the *Coordinate Reference System* association of the GM_Point that is the *origin*.

b) {offsetVectors → size = dimension} The number of offset vectors shall equal the dimension of the grid.

c) {offsetVectors->forAll(dimension = self.origin.CRS.theSC_CoordinateSystem.dimension)} The dimension of the offset vectors shall equal the dimension of the coordinate reference system, even if an offset vector is aligned with an axis of the external coordinate system.

8.10 CV_ReferenceableGrid

8.10.1 General

CV_ReferenceableGrid shall support two operations, coordTransform and invCoordTransform.

8.10.1.1 coordTransform

The operation *coordTransform (g: CV_GridCoordinate): DirectPosition* shall accept a CV_GridCoordinate as input and return a DirectPosition in the coordinate reference system identified through the association *Coordinate Reference System* (8.10.2). This is a coordinate transformation operation as defined by ISO 19111:2003.

8.10.1.2 invCoordTransform

The operation invCoordConv (p: DirectPosition): CV_GridCoordinate shall accept a DirectPosition in the coordinate reference system identified through the association *Coordinate Reference System* (8.10.2) as input and return the CV_GridCoordinate of the nearest CV_GridPoint. This is a coordinate transformation operation as defined by ISO 19111:2003.

8.10.2 Coordinate Reference System

The association *Coordinate Reference System* shall link the CV_ReferenceableGrid to the coordinate reference system to which it is referenceable. The association shall be navigable from *grid* to *crs*, but need not be navigable in the opposite direction.

8.11 CV_ContinousQuadrilateralGridCoverage

8.11.1 General

A CV_ContinuousQuadrilateralGridCoverage (Figure 18) is a subclass of CV_ContinuousCoverage that operates on a CV_GridValuesMatrix (8.14). The domain of a CV_ContinuousQuadrilateralGridCoverage is the convex hull of the collection of grid points defined by the CV_GridValuesMatrix. Evaluation of a CV_ContinuousQuadrilateralGridCoverage generates feature attribute values at direct positions within the convex hull of the CV_GridPoints provided by the CV_GridValuesMatrix. The general idea is to extend the coverage to direct positions within the interior of each grid cell by interpolation from the grid points at the corners of the cell.

8.11.2 interpolationType

The inherited attribute *interpolationType: CV_InterpolationMethod = bilinear* shall identify the interpolation method to be used by the operation *evaluate.*

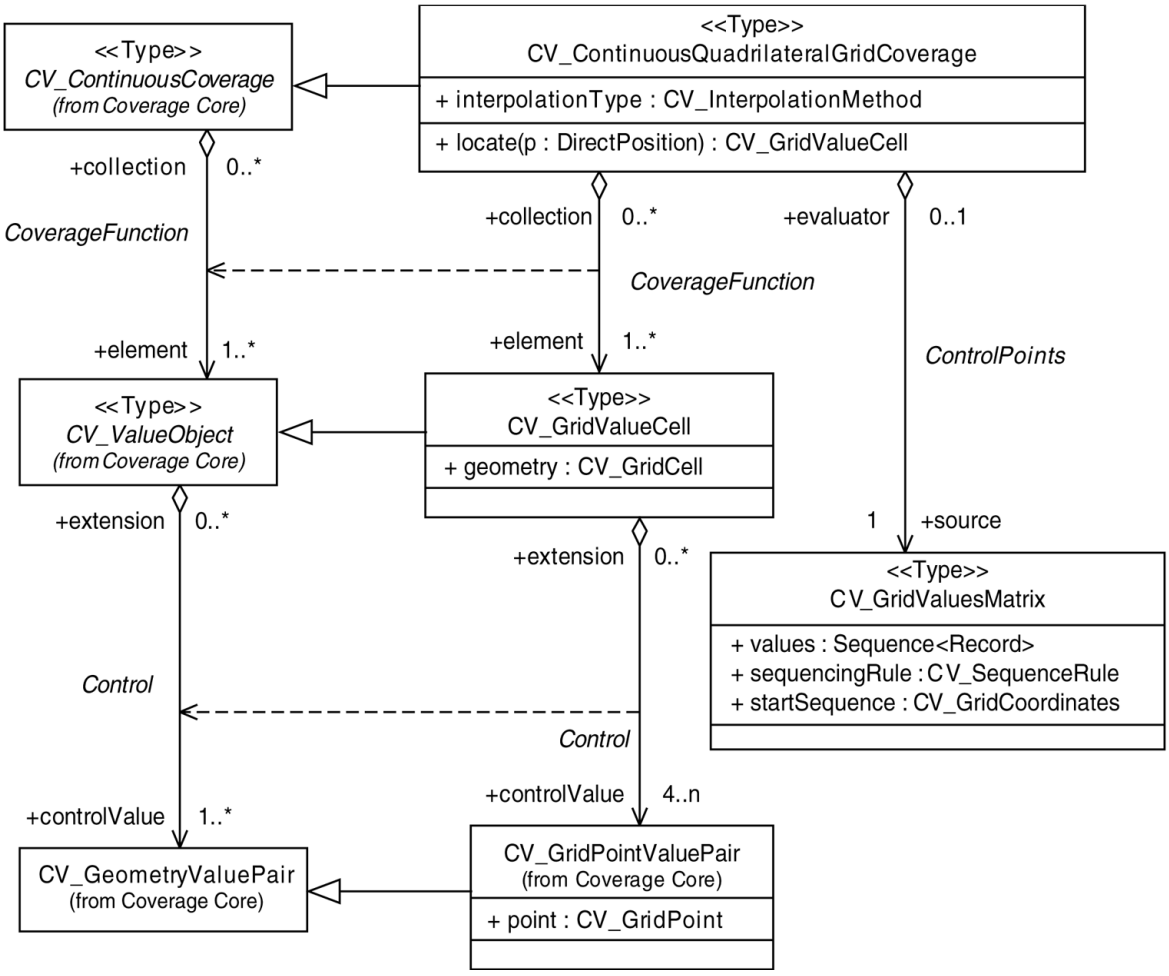


Figure 18 — CV_ContinuousQuadrilateralGridCoverage

8.11.3 locate

The operation *locate (p: DirectPosition): CV_GridValueCell* is inherited from CV_ContinuousCoverage with the restriction that it shall return a CV_GridValueCell. It shall accept a DirectPosition as input and return the CV_GridValueCell that contains that DirectPosition.

8.11.4 evaluate

The operation *evaluate (p: DirectPosition, list: Sequence <CharacterString>): Record* is inherited from CV_Coverage. Evaluation of a CV_ContinuousQuadrilateralGridCoverage involves two steps. The first is to use the information from the CV_ValuesMatrix (8.14) at CV_QuadrilateralGridCoverage.source to generate the CV_GridValueCell that contains the input DirectPosition; the second is to interpolate the feature attribute values at the DirectPosition from the CV_GridPointValuePairs at the corners of the CV_GridValueCell. Some interpolation methods (e.g. bicubic interpolation (C.8)) may require the use of CV_GridPointValuePairs outside of the CV_GridValueCell that contains the DirectPosition.

NOTE Nearest neighbour interpolation will return for any direct position within a CV_GridValueCell the Record associated with the CV_GridPointValuePair at the nearest corner of the CV_GridValueCell. In other words, a CV_Continuous GridCoverage that uses nearest neighbour interpolation acts as a discrete surface coverage.

8.11.5 ControlPoints

The association *ControlPoints* shall link this CV_ContinuousQuadrilateralGridCoverage to the CV_GridValuesMatrix that provides the data for the *evaluate* operation. The association shall be navigable from *evaluator* to *source*, but need not be navigable in the opposite direction.

8.11.6 CoverageFunction

The association *CoverageFunction* shall link this CV_ContinuousQuadrilateralGridCoverage to the set of CV_GridValueCells that provide the structure to support the *evaluate* operation.

8.12 CV_GridValueCell

8.12.1 General

CV_GridValueCell is a subclass of CV_ValueObject that supports interpolation within a CV_ContinuousQuadrilateralGridCoverage. A CV_GridValueCell is a collection of CV_GridPointValuePairs with a geometric structure defined by a CV_GridCell.

8.12.2 geometry

The attribute *geometry:CV_GridCell* shall hold the CV_GridCell that defines the structure of the CV_GridPointValuePairs that support the interpolation of a feature attribute value at a DirectPosition within the CV_GridCell.

8.12.3 Control

The association *Control* shall link the CV_GridValueCell to the CV_GridPointValuePairs (8.13) at its corners.

8.13 CV_GridPointValuePair

8.13.1 General

CV_GridPointValuePair is a subclass of CV_GeometryValuePair composed of a CV_GridPoint and a feature attribute value Record.

8.13.2 point

The attribute *point: CV_GridPoint* shall be the geometry member of the CV_GridPointValuePair. It shall be one of the CV_GridPoints linked to the CV_ValuesMatrix through the *Organization* association inherited from CV_Grid.

8.13.3 value

The attribute *value: Record* shall be the member of the CV_GridPointValuePair taken from the sequence *values* in the CV_GridValuesMatrix.

8.14 CV_GridValuesMatrix

8.14.1 General

CV_GridValuesMatrix (Figure 19) is a subclass of CV_Grid that ties feature attribute values to grid geometry. It has three attributes: values, sequencingRule and startSequence. It holds a sequence of records associated with a sequencing rule that specifies an algorithm for assigning records of feature attribute values to grid points.

8.14.1.1 values

The attribute *values: Sequence <Record>* shall be a sequence of N feature attribute value records where N is the number of grid points within the section of the grid specified by *extent*.

8.14.1.2 sequencingRule

The attribute *sequencingRule: CV_SequenceRule* shall describe how the grid points are ordered for association to the elements of the sequence *values*.

8.14.1.3 startSequence

The attribute *startSequence: CV_GridCoordinate* shall identify the grid point to be associated with the first record in the *values* sequence.

8.14.2 Constraints

The constraint {self.extent->notEmpty = True} indicates that a value shall be provided for the inherited attribute *extent* (8.3.4).

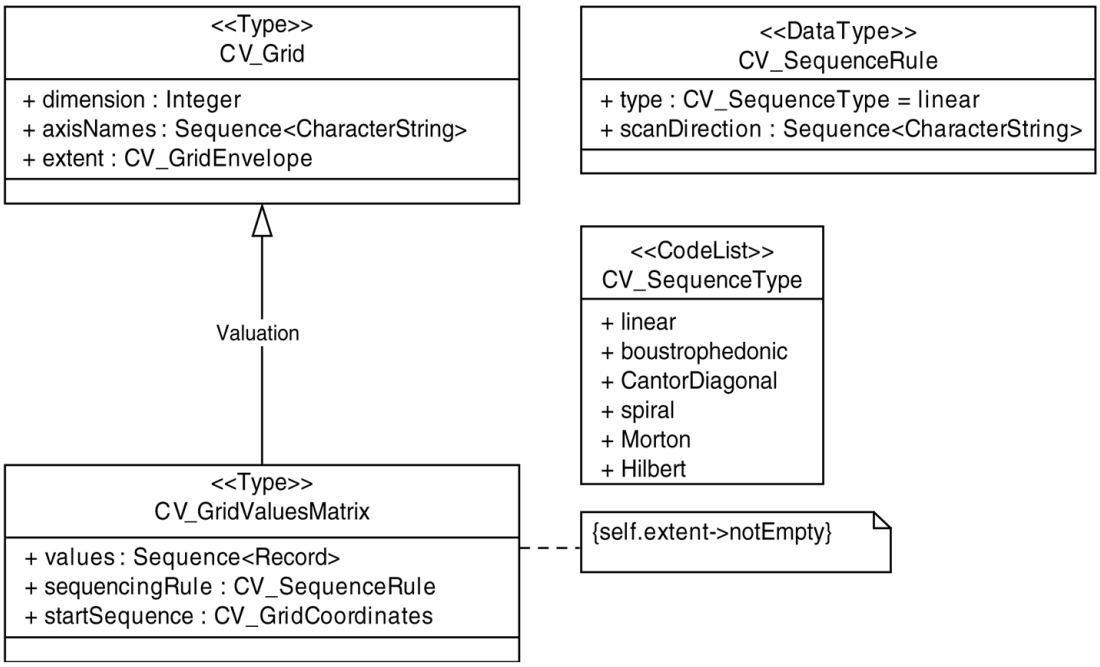


Figure 19 — CV_GridValuesMatrix

8.15 CV_SequenceRule

8.15.1 General

CV_SequenceRule is a data type that contains information for mapping grid coordinates to a position within the sequence of records of feature attribute values.

8.15.2 type

The attribute *type: CV_SequenceType* shall identify the type of sequencing method that shall be used. The default value shall be “linear”.

8.15.3 scanDirection

The attribute *scanDirection: Sequence <CharacterString>* shall be a list of signed axisNames that indicates the order in which grid points shall be mapped to position within the sequence of records of feature attribute values. An additional element may be included in the list to allow for interleaving of feature attribute values. See D.1 for more detailed information about scan directions.

8.16 CV_SequenceType

CV_SequenceType is a code list that identifies methods for sequential enumeration of the grid points. Methods for sequential enumeration are described in Annex D.

9 Hexagonal Grid Coverages

9.1 General

Coverages are sometimes based on tessellations composed of regular hexagons. Such tessellations are usually called hexagonal grids. In fact, the centers of a set of regular hexagons that form such a tessellation correspond to the grid points of a quadrilateral grid (Figure 20). That grid can be described as a rectified grid in which the two offset vectors are of equal length but differ in direction by 60°. The length of a side of the hexagon is L = S tan 30°, where S is the length of the offset vector. This means that the values in the coverage range can be stored as a grid values matrix (8.14) and accessed through a sequence rule (8.15). The hexagons are the Thiessen polygons that are generated around the grid points.

NOTE A set of Thiessen polygons generated from the grid points of any two-dimensional rectified grid described by two offset vectors that are equal in length but not orthogonal will be a set of congruent hexagons. The hexagons will be irregular unless the offset vectors differ in direction by exactly 60°.

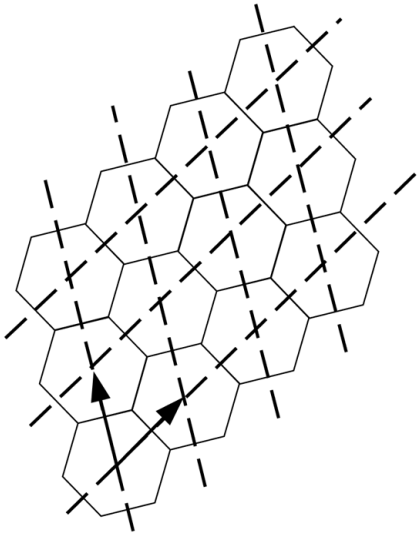


Figure 20 — A hexagonal grid

9.2 CV_HexagonalGridCoverage

9.2.1 General

A CV_HexagonalGridCoverage (Figure 21) evaluates a coverage at direct positions within a network of hexagons centred on a set of grid points. Evaluation is based on interpolation between the centres of the CV_ValueHexagons surrounding the input position.

9.2.2 interpolationType

The inherited attribute *interpolationType: CV_InterpolationMethod = "lost area"* shall identify the interpolation method to be used in evaluating the coverage. The most common interpolation methods are "lost area" (C.9) and "nearest neighbour" (C.2). Lost area interpolation can return a different Record of feature attribute values for each direct position within a CV_ValueHexagon. On the other hand, nearest neighbour interpolation will return for any direct position within a CV_ValueHexagon the Record associated with the CV_GridPointValuePair at the centre of the CV_ValueHexagon. In other words, a CV_HexagonalGridCoverage that uses nearest neighbour interpolation acts like a discrete surface coverage.

9.2.3 locate

The operation *locate (p: DirectPosition): CV_ValueHexagon* is inherited from CV_ContinuousCoverage with the restriction that it shall return a CV_ValueHexagon. It shall accept a DirectPosition as input and return the CV_ValueHexagon that contains that DirectPosition.

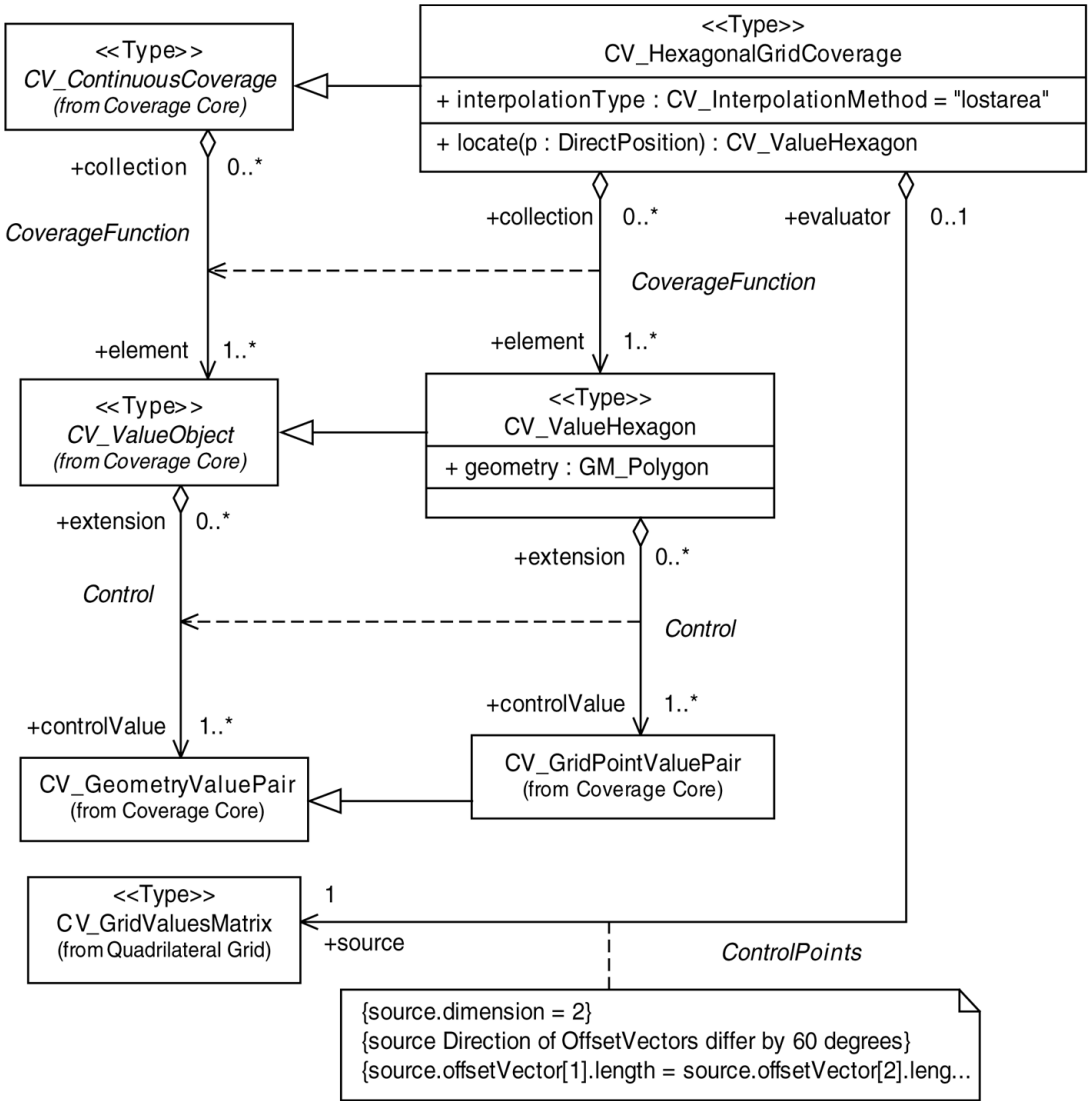


Figure 21 — CV_HexagonalGridCoverage

9.2.4 evaluate

The operation *evaluate (p: DirectPosition, list: Sequence <CharacterString>): Record* is inherited from CV_Coverage. Evaluation of a CV_HexagonalGridCoverage involves two steps. The first is to find the CV_ValueHexagon that contains the input DirectPosition; the second is to interpolate the feature attribute values at the DirectPosition from the CV_GridPointValuePairs at the centres of the surrounding CV_ValueHexagons.

9.2.5 CoverageFunction

The association *CoverageFunction* shall link this CV_HexagonalGridCoverage to the set of CV_ValueHexagons of which it is composed.

9.2.6 ControlPoints

The association *ControlPoints* shall link the CV_HexagonalGridCoverage to the CV_GridValuesMatrix for which it is an evaluator.

9.3 CV_GridValuesMatrix

CV_GridValuesMatrix is documented in 8.14, but is specialized by four constraints:

a) It is a CV_RectifiedGrid.

b) {source.dimension = 2} The inherited attribute *dimension* has a value of 2.

c) {source Direction of offsetVectors differ by 60 degrees} The offsetVectors differ in direction by 60 degrees.

d) {source,offsetVector[1].length = source.offsetVector[2].length} The lengths of the offsetVectors are equal.

9.4 CV_ValueHexagon

9.4.1 General

CV_ValueHexagon is a subclass of CV_ValueObject.

9.4.2 geometry

The attribute *geometry: GM_Polygon* shall hold the geometry of the CV_ValueHexagon centred on the CV_GridPointValuePair identified by the association *Control*.

9.4.3 Control

The association *Control* shall link this CV_ValueHexagon to the CV_GridPointValuePair at its centre.

10 Triangulated irregular network (TIN) coverages

10.1 General

The basic idea of a TIN is to partition the convex hull of the points in the domain of a discrete point coverage into a computationally unique set of non-overlapping triangles. Each triangle is formed by three of the points in the domain of the discrete point coverage. The Delaunay triangulation method is commonly used to produce TIN tessellations with triangles that are optimally equiangular in shape, and are generated in such a manner that the circumscribing circle containing each triangle contains no point of the discrete point coverage other than those at the vertices of the triangle (Figure 22). GM_TIN (ISO 19107:2003) describes a Delaunay triangulation.

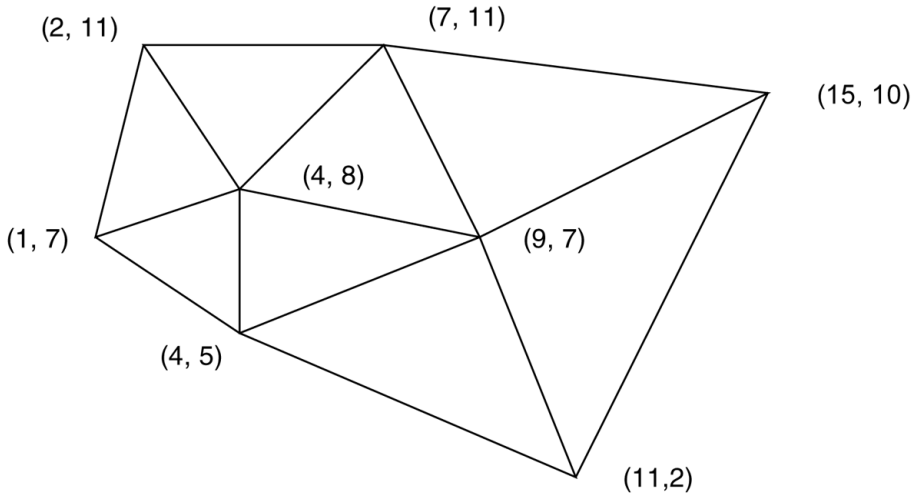


Figure 22 — An example of a triangulated irregular network with (x,y) coordinates

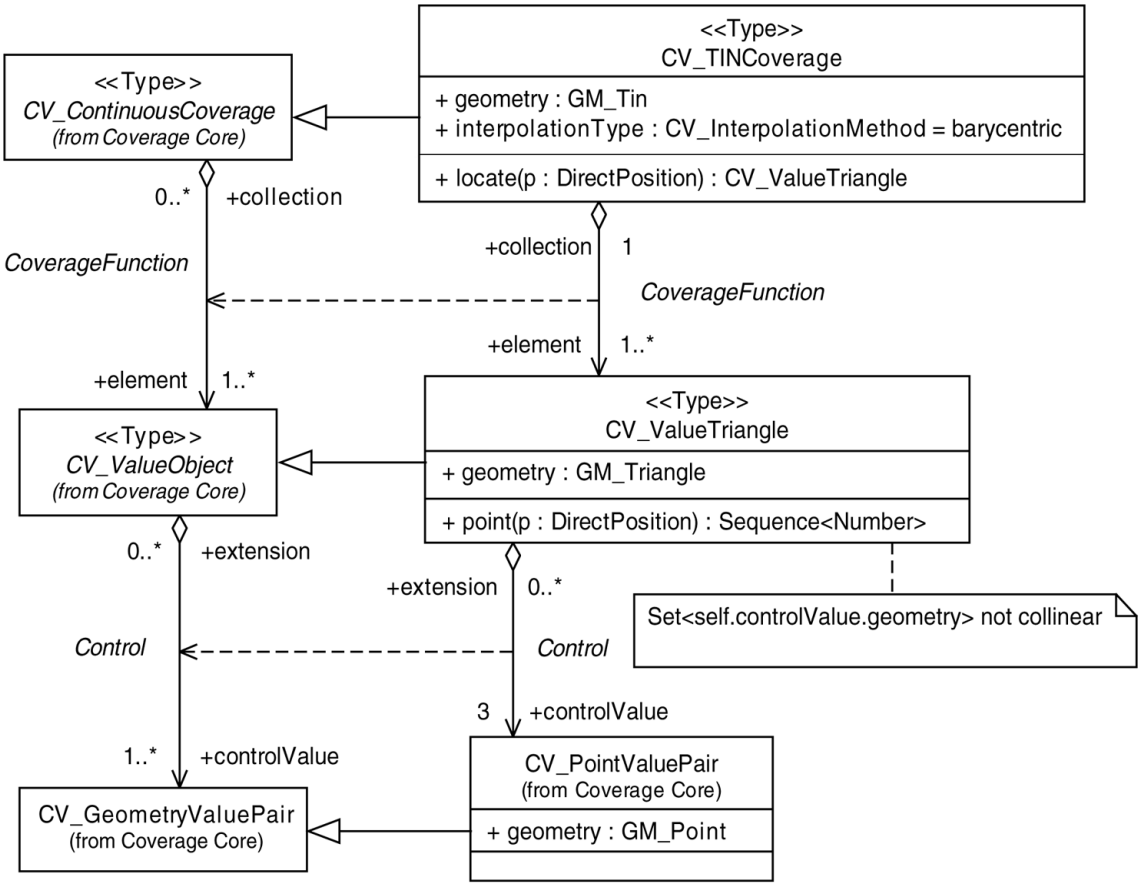


Figure 23 — CV_TINCoverage

10.2 CV_TINCoverage

10.2.1 General

A CV_TINCoverage (Figure 23) is a subclass of CV_ContinuousCoverage characterized by a GM_TIN. The feature attribute values are computed by interpolation within each triangle in the tessellation using the record of feature attribute values provided at each corner; that is, the feature attribute values are produced by an operation on CV_ValueTriangles.

10.2.2 geometry

The attribute *geometry: GM_TIN* shall hold the triangulated irregular network that provides the structure for evaluating the coverage. GM_TIN (ISO 19107:2003) includes a capability for using stop lines and break lines in the triangulation.

10.2.3 interpolationType

The inherited attribute *interpolationType: CV_InterpolationMethod = "barycentric"* shall specify the interpolation method to be used in evaluating the coverage. The most common interpolation method is "barycentric" (C.10).

10.2.4 locate

The operation *locate (p: DirectPosition): CV_ValueTriangle* is inherited from CV_ContinuousCoverage with the restriction that it shall return a CV_ValueTriangle. It shall accept a direct position as input and return the CV_ValueTriangle in which that direct position is located, together with the Records of feature attribute values assigned to the corners of that CV_ValueTriangle.

10.2.5 evaluate

The operation *evaluate (p: DirectPosition, list: Sequence <CharacterString>): Record* is inherited from CV_Coverage. Evaluation of a CV_TINCoverage involves two steps. The first is to find the CV_ValueTriangle that contains the input DirectPosition; the second is to interpolate the feature attribute values at the DirectPosition from the CV_PointValuePairs at the vertices of the CV_ValueTriangle.

10.2.6 CoverageFunction

The association *CoverageFunction* shall link this CV_TINCoverage to the CV_ValueTriangles of which it is composed.

10.3 CV_ValueTriangle

10.3.1 General

CV_ValueTriangle is a subclass of CV_ValueObject that consists of three CV_PointValuePairs where the GM_Points are non-collinear. CV_ValueTriangles are used for interpolation of a coverage.

10.3.2 geometry

The attribute *geometry:GM_Triangle* shall hold the GM_Triangle that defines the relative position of the three CV_PointValuePairs at its vertices.

10.3.3 point

The operation *point: (p: DirectPosition): Sequence <Number>* shall accept a direct position inside a CV_ValueTriangle and return the barycentric coordinates of the position as a sequence of numbers.

10.3.4 Control

The association *Control* shall link this CV_ValueTriangle to the three CV_PointValuePairs at its vertices.

10.3.5 Constraint

Set <self.controlValue.geometry> not collinear. The three GM_Points in the associated CV_PointValuePairs are not collinear.

11 Segmented curve coverages

11.1 General

Segmented curve coverages are used to model phenomena that vary continuously or discontinuously along curves, which may be elements of a network. The domain of a segmented curve coverage is described by a set of curves and includes all the direct positions in all of the curves in the set.

Arc-length parameterization of a curve simplifies interpolation between direct positions on the curve. Representation of phenomena that vary discontinuously along a curve requires segmentation of the curve into regions of continuous variation. Such segmentation is also simplified by arc-length parameterization. GM_Curve (ISO 19107:2003) supports arc-length parameterization. In particular, the operation:

GM_GenericCurve.parameterForPoint (p: DirectPosition):Set <Distance>, DirectPosition

returns the arc-length distance from the start point of the GM_Curve to the input DirectPosition.

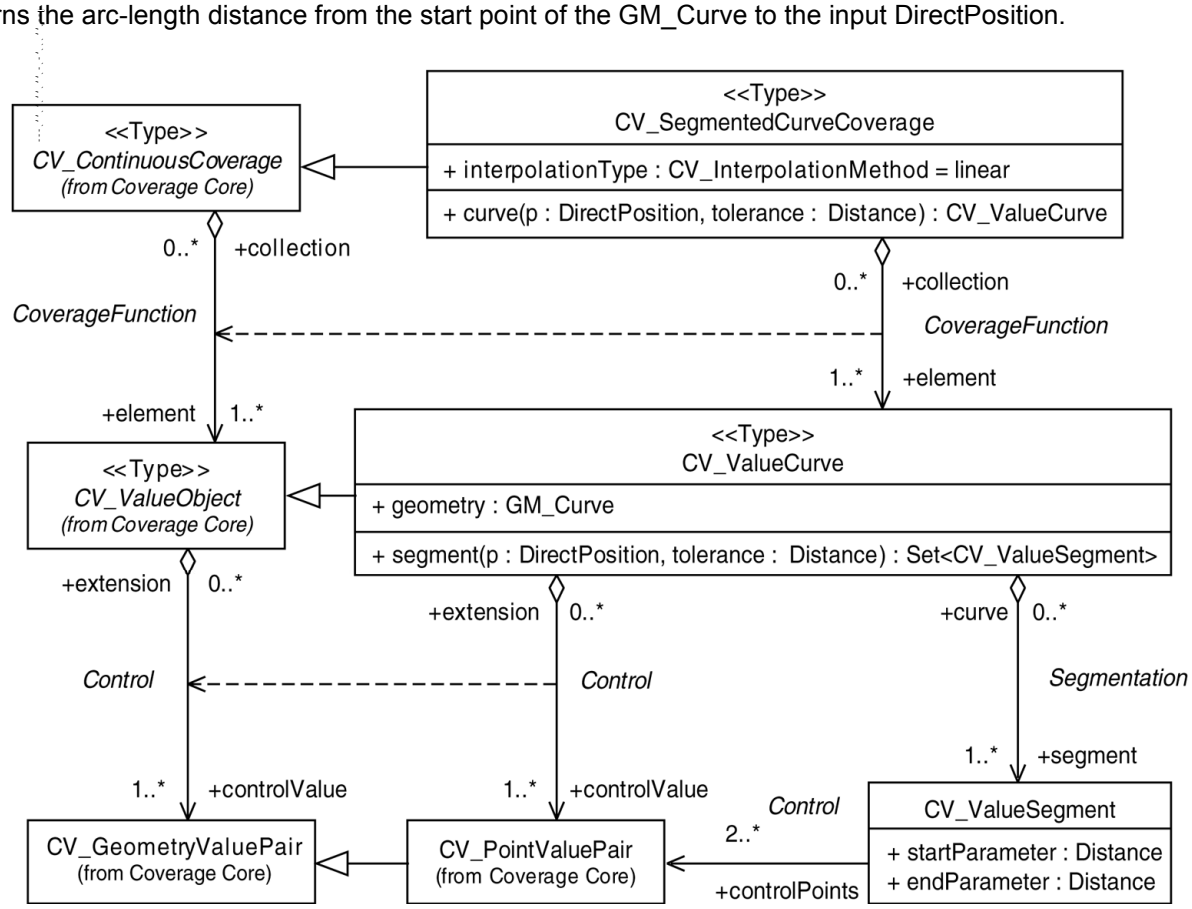


Figure 24 — SegmentedCurve Coverage

11.2 CV_SegmentedCurveCoverage

11.2.1 General

A CV_SegmentedCurveCoverage (Figure 24) operates on a domain composed of GM_Curves. A CV_SegmentedCurveCoverage is composed of a set of CV_ValueCurves, each of which maps feature attribute values to position on a GM_Curve.

11.2.2 interpolationType

The inherited attribute *interpolationType: CV_InterpolationMethod = "linear"* shall identify the interpolation method that shall be used to evaluate the coverage. The default value shall be "linear". An application schema may define other interpolation methods.

11.2.3 curve

The operation *curve (position: DirectPosition, tolerance: Distance = 0): CV_ValueCurve* shall accept a direct position as input and return the CV_ValueCurve nearest to that direct position. The operation shall return an error message if the direct position is not closed (i.e. within the distance specified by the *tolerance* parameter) to one of the CV_ValueCurves in the CV_SegmentedCurveCoverage. The default value for *tolerance* is zero.

11.2.4 CoverageFunction

The association *CoverageFunction* shall link this CV_SegmentedCurveCoverage to the *CV_*ValueCurves of which it is composed.

11.3 CV_ValueCurve

11.3.1 General

A CV_ValueCurve is composed of a GM_Curve with additional information that supports the determination of feature attribute values at any position on that curve. CV_ValueCurves depend upon the arc-length parameterization operations defined for GM_Curve in ISO 19107:2003.

11.3.2 geometry

The attribute *geometry: GM_Curve* shall be the GM_Curve that is the basis of this CV_ValueCurve.

11.3.3 segment

The operation *segment (position: DirectPosition): Set <CV_ValueSegment>* shall accept a DirectPosition as input and return the set of CV_ValueSegments nearest to that DirectPosition. This operation shall invoke the *parameterForPosition* operation defined for GM_Curve to obtain the Distance parameter corresponding to the input DirectPosition. The operation *parameterForPosition* returns the parameter value for the position on the GM_Curve closest to the input DirectPosition. In certain cases, the *parameterForPosition* may return more than one parameter value. The operation *segment* will normally return a single CV_ValueSegment. There are three cases for which it could return multiple CV_ValueSegments:

a) The CV_ValueCurve is not simple. The position on the curve that is closest to the input DirectPosition is a point of self-intersection. The operation *parameterForPoint* returns two or more parameter values. In this case, the operation *segment* shall raise an exception.

b) There are two or more positions on the CV_ValueCurve that are at the same minimal distance from the input DirectPosition. The operation *parameterForPoint* returns two or more parameter values. In this case, the operation *segment* shall raise an exception.

c) The position on the CV_ValueCurve that is closest to the input DirectPosition is at the end of one CV_ValueSegment and the start of the next. In this case, the operation shall return both CV_ValueSegments.

11.3.4 Segmentation

The association *Segmentation* shall link this CV_ValueCurve to the sequence of CV_ValueSegments of which it is composed.

11.3.5 Control

The association Control shall link this CV_ValueCurve to the set of two or more CV_PointValuePairs that provide control values for the interpolation along the CV_ValueCurve.

11.4 CV_ValueSegment

11.4.1 General

The limits of a CV_ValueSegment are specified by two values of the arc-length parameter of the GM_Curve underlying its parent CV_ValueCurve.

11.4.2 startParameter

The attribute *startParameter: Distance* shall be the value of the arc-length parameter of the parent curve at the start of this CV_ValueSegment.

11.4.3 endParameter

The attribute *endParameter: Distance* shall be the value of the arc-length parameter of the parent curve at the end of this CV_ValueSegment.

11.4.4 Control

The association *Control* shall link the CV_ValueSegment to the CV_PointValuePairs that provide control values for interpolation. Linear interpolation requires a minimum of two control values, usually those at the beginning and end of the CV_ValueSegment. Additional control values are required to support interpolation by higher order functions.

11.5 Evaluation

CV_SegmentedCurveCoverage inherits the operation *evaluate (p: DirectPosition, list: Sequence <CharacterString>): Record* from CV_Coverage. Evaluation of a CV_SegmentedCurveCoverage involves several steps:

1) Invoke the operation *curve* to find the CV_ValueCurve that contains the input DirectPosition.

2) Invoke the operation *CV_ValueCurve.segment* to find the CV_ValueSegment that contains the input DirectPosition.

3) If *CV_ValueCurve.segment* returns a single CV_ValueSegment, use the specified *interpolationType* to compute feature attribute values from the *associated* controlValues.

4) If *CV_ValueCurve.segment* returns a pair of conterminous CV_ValueSegments, compute the feature attribute values according to the rule specified by the attribute *commonPointRule* inherited from CV_Coverage.

Annex A
(normative)

Abstract test suite

A.1 Abstract tests for coverage interfaces

A.1.1 Simple coverage interface

The simple coverage interface test consists of the following:

a) Test Purpose: Verify that an application schema or profile instantiates CV_Coverage with the attribute *domainExtent*, the operation *evaluate*, and the associations *Domain, Range*, and *Coordinate Reference System*.

b) Test Method: Inspect the documentation of the application schema or profile.

c) Reference: ISO 19123, 5.3 and 5.4.

d) Test Type: Capability.

A.1.2 Discrete coverage interface

The discrete coverage interface test consists of the following:

a) Test Purpose: Verify that an application schema or profile satisfies the requirements of A.1; that it instantiates CV_DiscreteCoverage and its subtypes with the operations *locate, find* and *list*, and the association *CoverageFunction*; and that it instantiates the class CV_GeometryValuePair with the attributes *geometry* and *value*.

b) Test Method: Inspect the documentation of the application schema or profile.

c) Reference: ISO 19123, 5.7, 5.8 and 6.

d) Test Type: Capability.

A.1.3 Thiessen polygon coverage interface

The Thiessen polygon coverage interface test consists of the following:

a) Test Purpose: Verify that an application schema or profile satisfies the requirements of A.1 and that it instantiates the classes CV_ThiessenPolygonCoverage, and CV_ThiessenValuePolygon with their specified attributes, operations, associations and constraints.

b) Test Method: Inspect the documentation of the application schema or profile.

c) Reference: ISO 19123, Clause 7.

d) Test Type: Capability.

A.1.4 Quadrilateral grid coverage interface

The quadrilateral grid coverage interface test consists of the following:

a) Test Purpose: Verify that an application schema or profile satisfies the requirements of A.1 and that it instantiates the classes CV_Grid, CV_GridPoint, CV_GridCell, CV_GridValuesMatrix, CV_GridPointValuePair, CV_DiscreteGridPointCoverage, CV_ContinuousGridCoverage, and CV_GridValueCell with their specified attributes, operations, associations and constraints.

b) Test Method: Inspect the documentation of the application schema or profile.

c) Reference: ISO 19123, Clause 8.

d) Test Type: Capability.

A.1.5 Hexagonal grid coverage interface

The hexagonal grid coverage interface test consists of the following:

a) Test Purpose: Verify that an application schema or profile satisfies the requirements of A.1 and that it instantiates the classes CV_HexagonalGridCoverage, CV_ValueHexagon CV_Grid, CV_RectifiedGrid, CV_GridValuesMatrix, and CV_GridPointValuePair, with their specified attributes, operations, associations and constraints.

b) Test Method: Inspect the documentation of the application schema or profile.

c) Reference: ISO 19123, 8.3 – 8.13, Clause 9.

d) Test Type: Capability.

A.1.6 TIN coverage interface

The TIN coverage interface test consists of the following:

a) Test Purpose: Verify that an application schema or profile satisfies the requirements of A.1 and that it instantiates the classes CV_TINCoverage, and CV_ValueTriangle with their specified attributes, operations, associations and constraints.

b) Test Method: Inspect the documentation of the application schema or profile.

c) Reference: ISO 19123, Clause 10.

d) Test Type: Capability.

A.1.7 Segmented curve coverage interface

The segmented curve coverage interface test consists of the following:

a) Test Purpose: Verify that an application schema or profile satisfies the requirements of A.1 and that it instantiates the classes CV_SegmentedCurveCoverage, CV_ValueCurve, and CV_ValueSegment with their specified attributes, operations, associations and constraints.

b) Test Method: Inspect the documentation of the application schema or profile.

c) Reference: ISO 19123, Clause 11.

d) Test Type: Capability.

A.2 Abstract tests for coverage interchange

A.2.1 Discrete coverage interchange

The discrete coverage interchange test consists of the following:

a) Test Purpose: Verify that an interchange schema correctly implements the mandatory attributes and associations of CV_DiscreteCoverage or one of its subclasses.

b) Test Method: Inspect the documentation of the interchange schema.

c) Reference: ISO 19123, Clauses 5 and 6.

d) Test Type: Capability.

A.2.2 Thiessen polygon coverage interchange

The Thiessen polygon coverage interchange test consists of the following:

a) Test Purpose: Verify that an interchange schema correctly implements the mandatory attributes and associations of CV_ThiessenPolygonCoverage.

b) Test Method: Inspect the documentation of the interchange schema.

c) Reference: ISO 19123, Clause 7.

d) Test Type: Capability.

A.2.3 Quadrilateral grid coverage interchange

The quadrilateral grid coverage interchange test consists of the following:

a) Test Purpose: Verify that an interchange schema correctly implements the mandatory attributes and associations of CV_ContinuousQuadrilateralCoverage.

b) Test Method: Inspect the documentation of the interchange schema.

c) Reference: ISO 19123, Clause 8.

d) Test Type: Capability.

A.2.4 Hexagonal grid coverage interchange

The hexagonal grid coverage interchange test consists of the following:

a) Test Purpose: Verify that an interchange schema correctly implements the mandatory attributes and associations of CV_HexagonalGridCoverage.

b) Test Method: Inspect the documentation of the interchange schema.

c) Reference: ISO 19123, Clause 9.

d) Test Type: Capability.

A.2.5 TIN coverage interchange

The TIN coverage interchange test consists of the following:

a) Test Purpose: Verify that an interchange schema correctly implements the mandatory attributes and associations of CV_TINCoverage.

b) Test Method: Inspect the documentation of the interchange schema.

c) Reference: ISO 19123, Clause 10.

d) Test Type: Capability.

A.2.6 Segmented curve coverage interchange

The segmented curve coverage interchange test consists of the following:

a) Test Purpose: Verify that an interchange schema correctly implements the mandatory attributes and associations of CV_SegmentedCurveCoverage.

b) Test Method: Inspect the documentation of the interchange schema.

c) Reference: ISO 19123, Clause 11.

d) Test Type: Capability.

Annex B
(informative)

UML Notation

B.1 General

This annex provides a brief description of UML notation as used in the UML diagrams in this International Standard.

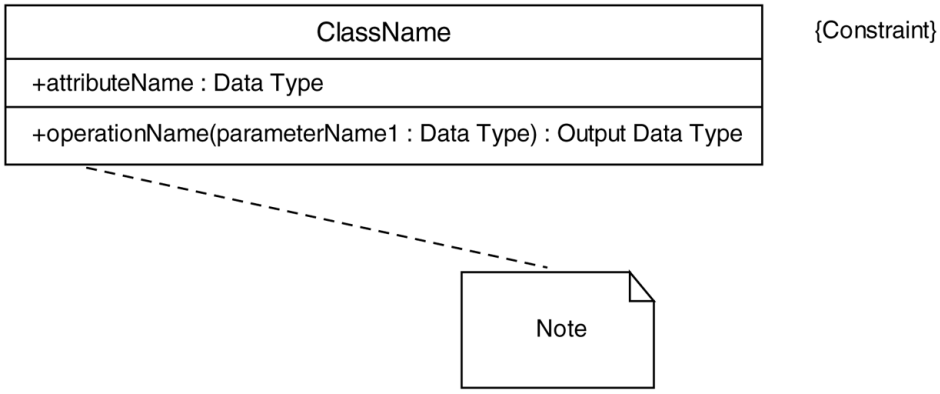


Figure B.1 — UML Class

B.2 Class

A UML class (Figure B.1) represents a concept within the system being modelled. It is a description of a set of objects that share the same attributes, operations, methods, relationships, and semantics. A class is drawn as a solid-outline rectangle with three compartments separated by horizontal lines. The top name compartment holds the class name and other general properties of the class (including stereotype); the middle list compartment holds a list of attributes; the bottom list compartment holds a list of operations. The attribute and operation compartments may be suppressed to simplify a diagram. Suppression does not indicate that there are no attributes or operations.

ISO/TS 19103 specifies that a class name shall include no blank spaces and that individual words in the name shall begin with capital letters.

B.3 Stereotype

Stereotypes extend the semantics, but not the structure of pre-existing types and classes. Class level stereotypes used in this International Standard include:

<<DataType>> A descriptor of a set of values that lack identity (independent existence and the possibility of side effects). Data types include primitive predefined types and user-definable types. A DataType is thus a class with few or no operations whose primary purpose is to hold the abstract state of another class for transmittal, storage, encoding or persistent storage.

<<Enumeration>> A data type whose instances form a list of named literal values. Both the enumeration name and its literal values are declared. Enumeration means a short list of well-understood potential values within a class. Classic examples are Boolean that has only two (or three) potential values: TRUE, FALSE (and NULL). Most enumerations will be encoded as a sequential set of Integers, unless specified otherwise. The actual encoding is normally only of use to the programming language compilers.

<<CodeList>> defined in ISO/TS 19103 is a flexible enumeration that uses string values through a binding of the Dictionary type key and return values as string types; e.g. Dictionary (String, String). Code lists are useful for expressing a long list of potential values. If the elements of the list are completely known, an enumeration shall be used; if the only likely values of the elements are known, a code list shall be used. Enumerated code lists may be encoded according to a standard, such as the ISO 3166-1. Code lists are more likely to have their values exposed to the user, and are therefore often mnemonic. Different implementations are likely to use different encoding schemes (with translation tables back to other encoding schemes available).

<<Type>> is a stereotype of class defined by ISO/IEC 19501. A Type is used to specify a domain of objects together with operations applicable to the objects without defining the physical implementation of those objects. It may also have attributes and associations that are defined solely for the purpose of specifying the behaviour of the type's operations and do not represent any actual implementation of state data.

B.4 Attribute

An attribute represents a characteristic common to the objects of a class. It is specified by a text string that can be parsed into elements that describe the properties of the attribute:

visibility name [multiplicity]: type-expression = initial-value

where:

visibility may be public (indicated by "+") or private (indicated by "–").

name is a character string. ISO/TS 19103 specifies that an attribute name shall include no blank spaces, that it shall begin with a lower case letter, and that individual words in the name, following the first word, shall begin with upper case letters.

multiplicity specifies the number of values that an instance of a class may have for a given attribute. Notation for multiplicity is explained in B.10.

type-expression identifies the data type of the attribute.

initial value specifies the default value for the attribute.

B.5 Operation

An operation represents a service that can be requested from an object. An operation is specified by a text string that can be parsed into elements that describe the properties of the operation:

visibility name (parameter-list): return-type-expression {property-string}

where:

visibility may be public (indicated by "+") or private (indicated by "–").

name is a character string. ISO/TS 19103 specifies that an operation name shall include no blank spaces, that it shall begin with a lower case letter, and that individual words in the name, following the first word, shall begin with upper case letters.

parameter-list is a list of parameters, each described by a parameter name and data type. These are assumed to be input parameters unless otherwise specified.

output parameter

return-type-expression identifies the data type of the value returned by the operation

{property-string} is an optional field that indicating property values that apply to the element

B.6 Constraint

A constraint specifies a semantic condition or restriction. Although ISO/IEC 19501 specifies an Object Constraint Language for writing constrains, a constraint may be written using any formal notation, or a natural language. A constraint is shown as a text string in braces { }. It is placed near the element to which it applies. If the notation for an element is a text string (such as an attribute), the constraint string may follow the element text string in braces. A constraint included as an element in a list applies to all subsequent elements in the list, down to the next constraint element or the end of the list.

B.7 Note

A note contains textual information. It is shown as a rectangle with a "bent corner" in the upper right corner, attached to zero or more model elements by a dashed line. Notes may be used to contain comments or constraints.

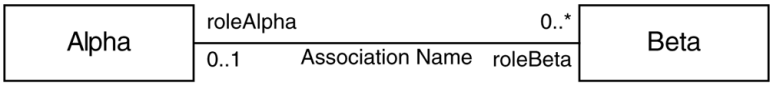


Figure B.2 — UML Associations

B.8 UML Associations

An association (Figure B.2) is a semantic relationship between classes that specifies connections between their instances. An association is drawn as a solid line connecting to class rectangles. An association may have a name, represented as a character string placed near the line, but not close to either end. ISO/TS 19103 specifies that an attribute name shall include no blank spaces and that individual words in the name shall begin with upper case letters. The association ends are adorned with information pertinent to the class at that end of the association, including multiplicity and role name.

B.9 Role name

A role name adorning an association end specifies behaviour of the class at that end with respect to the class at the other end of the association. In Figure B.2, roleAlpha describes the role that the class named Alpha has with respect to the class named Beta. A role name is represented as a Character String. ISO/TS 19103 specifies that a role name shall include no blank spaces, that it shall begin with a lower case letter, and that individual words in the name, following the first word, shall begin with upper case letters.

B.10 Multiplicity

Multiplicity specifies the number of instances of a class that may be associated with a class at the other end of the association. The values shown in Figure B.3 are all valid. They have the following meanings:

- zero or one instance of Alpha may be associated with one instance of Beta,
- zero or more instances of Beta may be associated with one instance of Alpha,
- one and only one instance of Gamma may be associated with one instance of Delta,
- n being an integer number, n and only n instances of Delta may be associated with one instance of Gamma,

- n 1 and n 2 being integer numbers, with n 2 > n 1, the number of instances of Epsilon that may be associated with an instance of Phi may be within the range n 1 to n 2,
- n being an integer number, n or more instances of Phi may be associated with one instance of Epsilon.

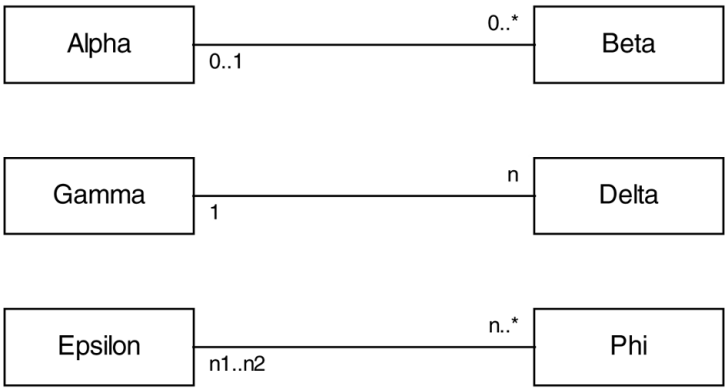


Figure B.3 — UML Multiplicity

B.11 Navigability

An arrow may be attached to the end of an association path to indicate that navigation is supported toward the class attached to the arrow. For example, in Figure B.4, the association is navigable from user to supplier. This means that an instance of the class Phi has access to information held in an instance of the class Epsilon. For example, an operation specified for Epsilon might use the value of an attribute of Phi.

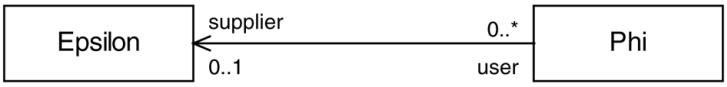


Figure B.4 — UML Navigability

B.12 Aggregation

Associations may be used to show aggregation or composition relationships between classes. An open diamond on an association end indicates that the class at that end of the association is an aggregate of instances of the class at the other end of the association. For example, the class named Gamma in Figure B.5 is an aggregate of zero or more instances of the class named Delta. Aggregation is considered a weak form of composition. The members of an aggregation can exist independently of the aggregation, and can be members of more than one aggregation.

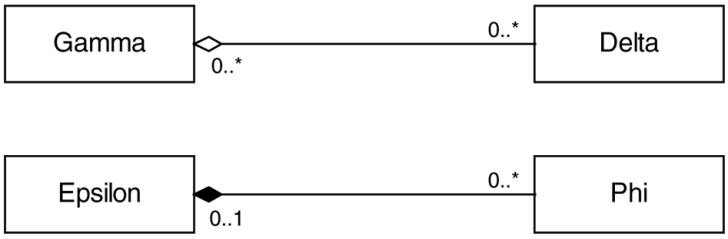


Figure B.5 — Aggregation and Composition

B.13 Composition

A closed diamond on an association end indicates that the class at that end of the association is composed of instances of the class at the other end of the association. For example, the class named Epsilon in Figure B.5 is composed of zero or more instances of the class named Phi. Members of a composite cannot exit independently of the composite class, nor can they be members of more than one composite class.

B.14 Dependency

A dependency states that the implementation or functioning of one or more elements requires the presence of one or more other elements. It indicates a semantic relationship between two model elements (or two sets of model elements). It relates the model elements themselves and does not require a set of instances for its meaning. A dependency is shown as a dashed arrow between two model elements. The model element at the tail of the arrow (the client) depends on the model element at the arrowhead (the supplier). The kind of dependency may be indicated by a keyword in guillemets, such as <<import>>, <<refine>>, or <<use>>. In the example of Figure 8, Epsilon has a <<use>> dependency upon Phi.

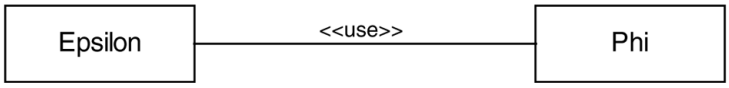


Figure B.6 — Dependency

B.15 Generalization

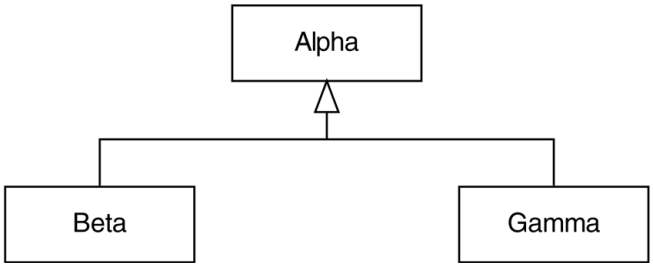


Figure B.7 — UML Generalization

ISO/IEC 19501 defines generalization (Figure B.7) as a taxonomic relationship between a more general element and a more specific element. The more specific element is fully consistent with the more general element and contains additional information. An instance of the more specific element may be used where the more general element is allowed. Generalization is shown as a solid-line path from the child (the more specific element, such as a subclass) to the parent (the more general element, such as a superclass), with a large hollow triangle at the end of the path where it meets the more general element.

Annex C
(informative)

Interpolation methods

C.1 General

Evaluation of a continuous coverage involves interpolation between known feature attribute values associated with geometric objects in the domain of the discrete coverage that is provided as control for the continuous coverage. The CodeList CV_InterpolationMethod (5.9.2) includes nine interpolation methods. Each is used in the context of specified geometric configurations (Table C.1). Since CV_InterpolationMethod is a CodeList, it may be extended in an application schema that specifies additional interpolation methods.

Table C.1 — Interpolation methods

Method	Coverage Type	Value Object Dimension	Subclause
Nearest Neighbour	Any	Any	C.2
Linear	Segmented Curve	1	C.3
Quadratic	Segmented Curve	1	C.4
Cubic	Segmented Curve	1	C.5
Bilinear	Quadrilateral Grid	2	C.6
Biquadratic	Quadrilateral Grid	2	C.7
Bicubic	Quadrilateral Grid	2	C.8
Lost Area	Thiessen Polygon, Hexagonal Grid	2	C.9
Barycentric	TIN	2	C.10

C.2 Nearest neighbour interpolation

Nearest neighbour interpolation can be applied to any coverage. It generates a feature attribute value at a direct position by assigning it the feature attribute value associated with the nearest domain object in the domain of the coverage. Nearest neighbour interpolation extends a discrete coverage to a step function defined on the convex hull of the domain objects in the domain of the coverage. Nearest neighbour interpolation is the only interpolation method described in this International Standard that can be used to interpolate attributes that have nominal or ordinal values.

C.3 Linear interpolation

Linear interpolation is commonly used to interpolate along value curves (11.3). It is based on the assumption that feature attribute values vary in proportion to distance along a value segment:

$$v = a + bx$$

Linear interpolation may be used to interpolate feature attribute values along a line segment connecting any two point value pairs. It may also be used to interpolate feature attribute values at positions along a curve of any form, if the positions are described by values of an arc-length parameter.

Given two point value pairs (p_s, v_s) and (p_t, v_t), where p_s is the start point and p_t is the end point of a value segment, and v_s and v_t are the feature attribute values associated with those points, the feature attribute value v_i associated with the direct position p_i is:

$$v_i = v_s + (v_t - v_s)\ ((p_i - p_s)/(p_t - p_s))$$

NOTE In the case of a discrete point coverage, the "steps" of the step function are the Thiessen polygons generated by the set of points in the domain of the coverage.

C.4 Quadratic interpolation

Quadratic interpolation is also used to interpolate along curves. It is based on the assumption that feature attribute values vary as a quadratic function of distance along a value segment:

$$v = a + bx + cx^2$$

where a is the value of a feature attribute at the start of a value segment and v is the value of a feature attribute at distance x along the curve from the start. Three point value pairs are needed to provide control values for calculating the coefficients of the function.

C.5 Cubic interpolation

Cubic interpolation is also used to interpolate along curves. It is based on the assumption that feature attribute values vary as a cubic function of distance along a value segment:

$$v = a + bx + cx^2 + dx^3$$

where a is the value of a feature attribute at the start of a value segment and v is the value of a feature attribute at distance x along the curve from the start. Four point value pairs are needed to provide control values for calculating the coefficients of the function.

C.6 Bilinear interpolation

Bilinear interpolation is used to interpolate feature attribute values at direct positions within a quadrilateral grid. It is based on the assumption that feature attribute values vary as a bilinear function of position within the grid cell:

$$v = a + bx + cy + dxy$$

Given a direct position, p, contained in a grid cell whose vertices are V, $V + V_1$, $V + V_2$, and $V + V_1 + V_2$, where V_1 and V_2 are the offset vectors of the grid, and with feature attribute values at these vertices of v_1, v_2, v_3, and v_4, respectively, there are unique numbers i and j, with $0 \leqslant i < 1$, and $0 \leqslant j < 1$ such that $p = V + iV_1 + jV_2$. The feature attribute value at p is:

$$v = (1\text{-}i)(1\text{-}j)\ v_1 + i(1\text{-}j)\ v_2 + j(1\text{-}i)\ v_3 + ij\ v_4$$

NOTE In an unrectified grid, V_1 and V_2 are the unit vectors (0,1) and (1,0).

C.7 Biquadratic interpolation

Biquadratic interpolation is also used to compute feature attribute values at direct positions within a quadrilateral grid. It is based on the assumption that feature attribute values vary as a biquadratic function of position within the grid cell:

$$v = a + bx + cy + dx^2 + exy + fy^2 + gx^2y + hxy^2 + ix^2y^2$$

See [6] for a discussion of algorithms for implementing biquadratic interpolation.

C.8 Bicubic interpolation

Bicubic interpolation is also used to compute feature attribute values at direct positions within a quadrilateral grid. It is based on the assumption that feature attribute values vary as a bicubic function of position within the grid cell:

$$v = a_0 + a_1x + a_2y + a_3x^2 + a_4xy + a_5y^2 + a_6x^2y + a_7\,x\,y^2 + a_8\,x^2\,y^2 + a_9\,x^3 + a_{10}\,y^3 + a_{11}\,x^3\,y + a_{12}\,x\,y^3 + a_{13}\,x^3\,y^2 + a_{14}\,x^2\,y^3 + a_{15}\,x^3\,y^3$$

See [8] for a discussion of algorithms for implementing bicubic interpolation.

C.9 Lost area interpolation

Lost area interpolation extends a discrete point coverage to a continuous function, f, defined on the convex hull of the domain of the point coverage.

Let $\{p_1, p_2, \ldots, p_n\}$ be the domain of the point coverage, and let $\{T_1, T_2, \ldots, T_n\}$ be the Thiessen polygons generated by the set $\{p_1, p_2, \ldots, p_n\}$.

Suppose it is desired to calculate $f(q)$, where q is a direct position in the convex hull of $\{p_1, p_2, \ldots, p_n\}$. Begin by forming the Thiessen polygons generated by $\{p_1, p_2, \ldots, p_n\}$; then add p to the set $\{p_1, p_2, \ldots, p_n\}$, and form the Thiessen polygons for the set of n+1 points: $\{p_1, p_2, \ldots, p_n, q\}$. The two sets of polygons are identical, except that each of the polygons coterminous with the polygon containing q "loses area" to the new polygon containing q.

The interpolation forms the weighted average such that each feature attribute value contributes to the feature attribute value at p according to the amount of area its polygon lost to the polygon at q. More formally:

a) Suppose that the discrete point coverage is characterized by the point value pairs: $\{(p_1, v_1), (p_2, v_2), \ldots, (p_n, v_n)\}$.

b) Among the Thiessen polygon set formed by $\{p_1, p_2, \ldots, p_n, q\}$, those coterminous with the polygon containing q are $\{T_1, T_2, \ldots, T_k\}$.

c) The corresponding Thiessen polygons from the set generated by $\{p_1, p_2, \ldots, p_n\}$ are $\{T'_1, T'_2, \ldots, T'_k\}$.

d) The area lost by the i^{th} polygon is $T'i - Ti$.

e) The total area lost is $\Sigma\,(T'i - Ti)$ where the sum is over i from 1 to k (that is, the sum is over all polygons that lost area to the polygon containing q). Note that this sum is the same as the area of the Thiessen polygon containing p.

f) Then the interpolated feature attribute value at q is:

$f(q) = (\Sigma vi * (T'i - Ti)) / \Sigma\,(T'i - Ti)$ where the summations are over the same range: $i = 1, \ldots, k$.

C.10 Barycentric interpolation

Let P, Q, and R denote the vertices of a triangle. For any direct position, S, in the triangle, there is a unique triple of numbers, i, j, and k, with $0 \leqslant i \leqslant 1$, $0 \leqslant j \leqslant 1$, and $0 \leqslant k \leqslant 1$, and with $i + j + k = 1$, such that

$$S = iP + jQ + kR$$

The numbers (i, j, k) are the barycentric coordinates of S.

Given a value triangle composed of the CV_PointValuePairs (p_1, v_1), (p_2, v_2), and (p_3, v_3), and a direct position, S, inside it, the barycentric coordinates of S are (i, j, k), where $S = ip_1 + jp_2 + kp_3$ and the feature attribute value at S is $v = iv_1 + jv_2 + kv_3$.

NOTE The name "barycentric" comes from the fact that using the equation above, S is the centre of mass of a triangle with point masses of size i, j, and k at the corners P, Q and R respectively. As one allocates mass to the three corners, the centre of mass can occupy any direct position in the triangle. For details, see [5].

Annex D
(informative)

Sequential enumeration

D.1 General

A sequential enumeration specifies the order in which attribute value records are assigned to grid points. CV_SequenceType (8.16) provides a list of codes for identifying types of sequencing methods. This annex explains those types in greater detail.

There are several sequencing rules based on incrementing – or decrementing – grid coordinate values in a simple fashion. More complex space filling curves can also be used. Space filling curves are generated by progressively subdividing a space in a regular way and connecting the elements resulting from each subdivision according to some rule. They can be used to generate a grid, but they can also be used to assign an ordering to the grid points or grid cells in a separately defined grid. They lend themselves more readily than simple incrementing methods to sequencing in grids that have irregular shapes or cells of variable size.

In every case, ordering of the grid cells starts by incrementing coordinates along one grid axis. At some point in the process, it begins to increment coordinates along a second grid axis, then a third, and so on until it has progressed in the direction of each of the grid axes. The figures in this annex provide examples. The attribute *CV_SequenceRule.scanDirection* (8.15.3) provides a list of signed axis names that identifies the order in which scanning takes place. The list may include an additional element to support interleaving of feature attribute values (see C.8 for a more detailed discussion of interleaving).

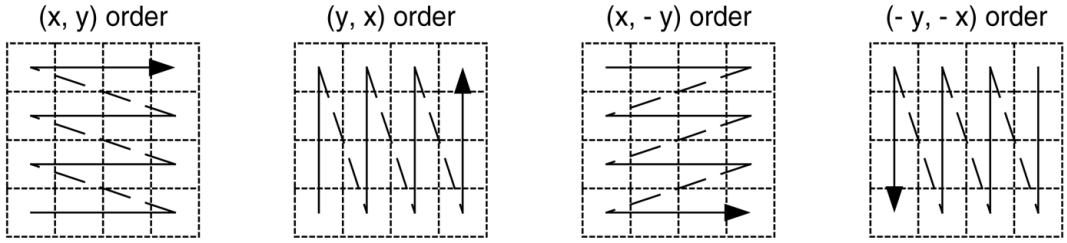
Ordering is continuous if consecutive pairs of grid cells in the sequence are maximally connected. It is semi-continuous if consecutive pairs of grid cells are connected, but less than maximally connected, and discontinuous if consecutive pairs of cells are not connected.

EXAMPLE In the 2-dimensional case, a cell is connected to the eight cells with which it shares at least one corner. It is maximally connected to the four cells with which it shares an edge and two corners. In the three dimensional case, a cell is maximally connected to those cells with which it shares a face.

NOTE In the example diagrams of this annex, continuous segments of scan lines are shown as solid lines, and discontinuous segments are shown as dashed lines.

D.2 Linear scanning

In linear scanning (Figure D.1), feature attribute value records are assigned to consecutive grid points along a single grid line parallel to the first grid axis listed in scanDirection. Once scanning of that row is complete, assignment of feature attribute value records steps to another grid line parallel to the first, and continues to step from grid line to grid line in a direction parallel to the second axis. If the grid is 3-dimensional, the sequencing process completes the assignment of feature attribute value records to all grid points in one plane, then steps to another plane, then continues stepping from plane to plane in a direction parallel to the third axis of the grid. The process can be extended to any number of axes. Linear scanning is continuous only along a single grid line.



NOTE The axes of 2-dimensional grids are often called “row” (horizontal) and “column” (vertical). In this case, scanning in (x,y) order is sometimes called row or row-major scanning.

Figure D.1 — Examples of linear scanning in a 2-dimensional grid

D.3 Boustrophedonic scanning

In a variant of linear scanning, known as boustrophedonic or byte-offset scanning, the direction of the scan is reversed on alternate grid lines (Figure D.2). In the case of a 3-dimensional grid, it will also be reversed in alternate planes. Boustrophedonic scanning is continuous.

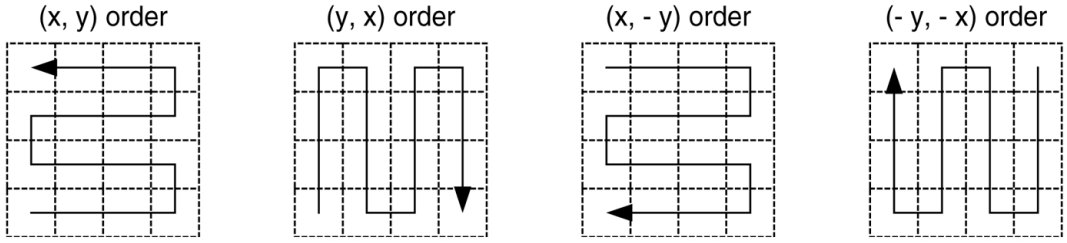


Figure D.2 — Examples of boustrophedonic scanning in a 2-dimensional grid

D.4 Cantor-diagonal scanning

Cantor-diagonal scanning, also called zigzag scanning, orders the grid points in alternating directions along parallel diagonals of the grid (Figure D.3). The scan pattern is affected by the direction of first step. Like linear scanning, Cantor-diagonal scanning can be extended to grids of three or more dimensions by repeating the scan pattern in consecutive planes. Cantor-diagonal scanning is semi-continuous within a single plane.

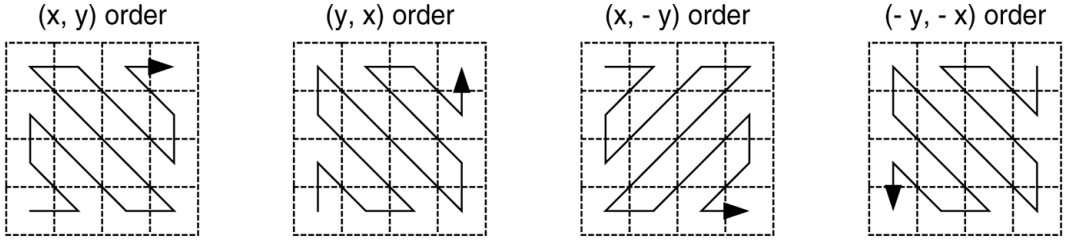


Figure D.3 — Examples of Cantor-diagonal scanning in a 2-dimensional grid

D.5 Spiral scanning

Spiral scanning (Figure D.4) can begin either at the centre of the grid (outward spiral), or at a corner (inward spiral). Like linear or Cantor-diagonal scanning, spiral scanning can be extended to grids of three or more dimensions by repeating the scan pattern in consecutive planes. Spiral scanning is continuous in any one plane, but continuity in grids of more than two dimensions can only be maintained by reversing the inward/outward direction of the scan in alternate planes.

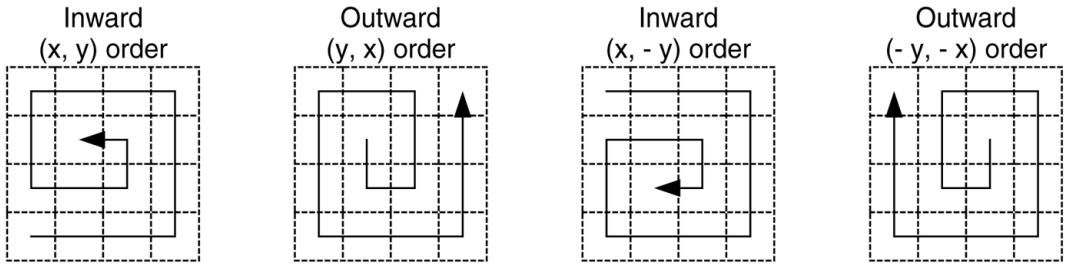


Figure D.4 — Examples of spiral scanning in a 2-dimensional grid

D.6 Morton order

Morton ordering is based on a space-filling curve generated by progressively subdividing a space into quadrants and ordering the quadrants in a Z pattern as shown in Figure D.5. The ordering index for each grid point is computed by converting the grid coordinates to binary numbers and interleaving the bits of the resulting values. Given the list of the grid axes specified by *CV_SequenceRule.scanDirection,* the bits of the coordinate corresponding to an axis are less significant than those of the coordinate corresponding to the next axis in the list. Morton ordering can be extended to any number of dimensions. Morton ordering is discontinuous.

NOTE Because of the shape of the curve formed by the initial ordering of quadrants, Morton ordering is also known as Z ordering.

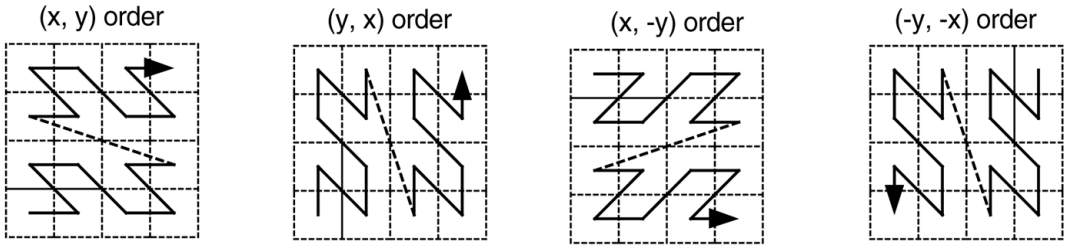


Figure D.5 — Examples of Morton ordering in a 2-dimensional grid

A grid generated with the Morton ordering technique will be square and its size in each direction will be a multiple of a power of two. However, the bit interleaving technique for generating an index can be used to order the grid points in any grid, including grids that are irregular in shape or have grid cells of different sizes (Figure D.6).

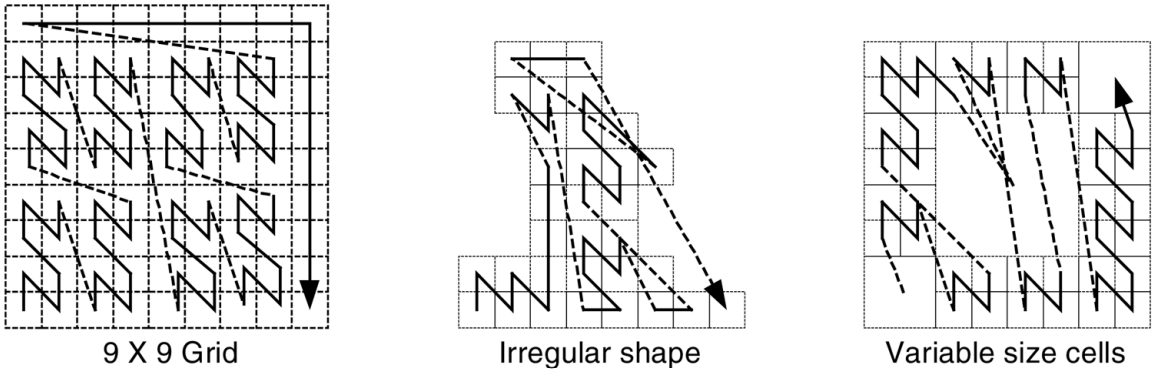


Figure D.6 — Examples of Morton ordering in irregular grids

D.7 Hilbert order

Like Morton ordering, Hilbert ordering is based on a space-filling curve generated by progressively subdividing a space into quadrants, but the initial pattern of subdivision is different for Hilbert curves. Further subdivision involves replacement of parts of the curve by different patterns (Figure D.7), unlike the simple replication of a single pattern as in Morton ordering. There are two sets of patterns. The left-hand column of the figure includes those for which the sense of the scan directions is the same – both are positive or both negative. The right-hand column of the figure includes those for which the sense of the scan directions is opposite – one is positive and one is negative. A Hilbert curve can only be constructed with patterns from the same set; it uses all the patterns in that set.

NOTE Because of the shape of the curve formed by the initial ordering of quadrants, Hilbert ordering is also known as Pi ordering.

Computation of the ordering index is more complicated for Hilbert ordering than for Morton ordering. Algorithms for the 2-dimensional case (Figure D.8) are described in [7] and [8]. 3-dimensional Hilbert curves are discussed in [9]. Hilbert ordering is continuous.

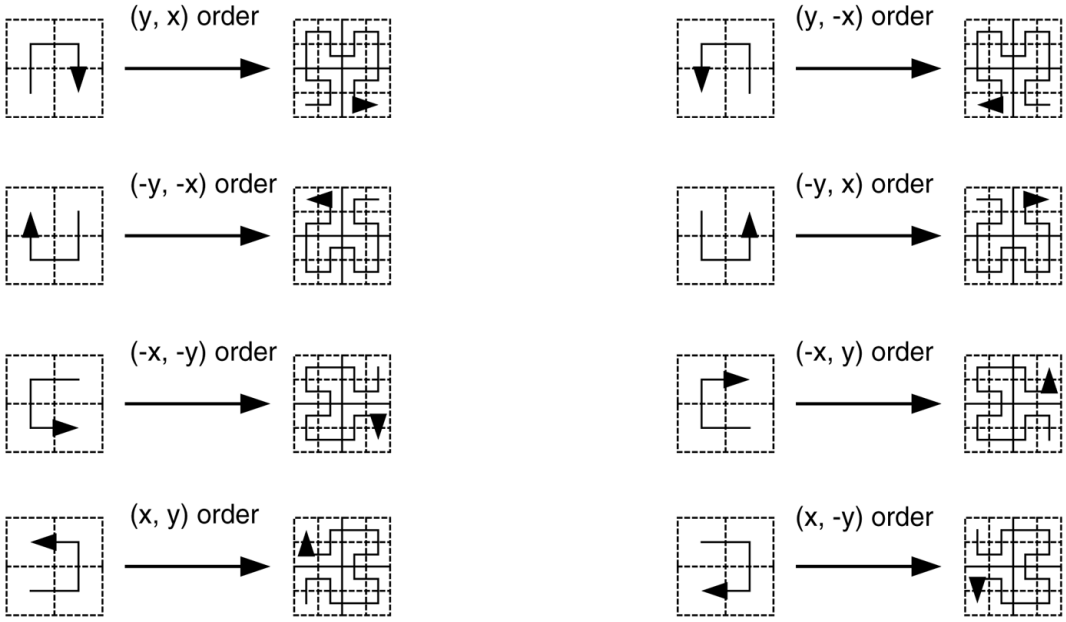


Figure D.7 — Replacement patterns for generating a Hilbert curve

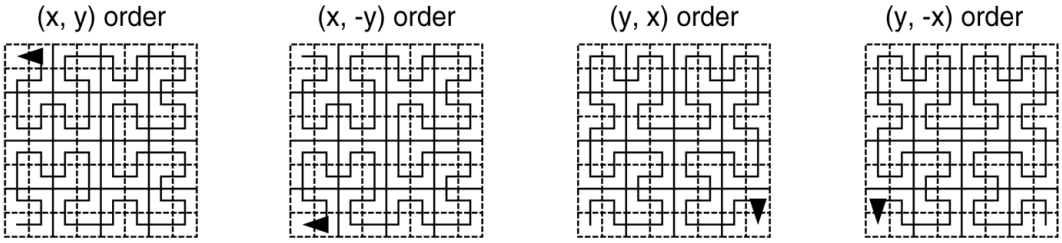


Figure D.8 — Examples of Hilbert ordering in a 2-dimensional grid

D.8 Interleaving of feature attribute values

When the range of a grid coverage includes more than one feature attribute, the feature attribute values may be interleaved in various ways within a list. Such interleaving can be described by including an element for the range in the list of axes provided by the *scanDirection* attribute of CV_SequenceRule. The index for the record of attributes is then incremented in the same way as the coordinates.

EXAMPLE Consider the 2 × 2 grid in Figure D.9. It has a range (r) of two attributes, A and B. Assuming a linear scan positive first in the x and then in the y direction, the scan order can be selected to access the feature attribute values in the different ways shown in Table D.1.

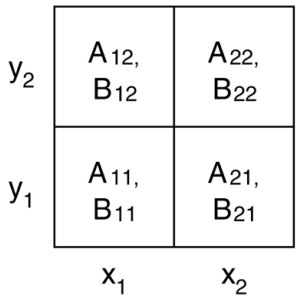


Figure D.9 — Example of a 2-dimensional grid with a range of two attributes

Table D.1 — Examples of interleaving

Order	Scan direction		
	r, x, y	**x, y, r**	**x, r, y**
1	A_{11}	A_{11}	A_{11}
2	B_{11}	A_{21}	A_{21}
3	A_{21}	A_{12}	B_{11}
4	B_{21}	A_{22}	B_{21}
5	A_{12}	B_{11}	A_{12}
6	B_{12}	B_{21}	A_{22}
7	A_{22}	B_{12}	B_{12}
8	B_{22}	B_{22}	B_{22}

Bibliography

[1] Open Geospatial Consortium, *The OpenGIS Specification: Abstract Specification*, Wayland, Massachusetts, 1999. Available at <http://www.opengeospatial.org/specs/>

[2] *Dictionary of Computing*, Fourth Edition, Oxford University Press, 1996

[3] ISO 19101:2002, *Geographic information — Reference model*

[4] ISO/IEC 19501:2005, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*

[5] HILTON, P. J. and WYLIE, S., *Homology Theory, an Introduction to Algebraic Topology*, Cambridge University Press, 1960

[6] KIDNER, D.B., *Higher-order interpolation of regular grid digital elevation models*, International Journal of Remote Sensing Vol. 24, No. 14, July 2003, pp. 2981-2987

[7] LAURINI, R. and THOMPSON, D., *Fundamentals of Spatial Information Systems,* Academic Press, 1992

[8] GOODCHILD, M. F. and GRANDFIELD, A. W., *Optimizing Raster Storage: An Examination of Four Alternatives,* IAuto-Carto 6, 1983

[9] GILBER, W., *A Cube-filling Hilbert Curve*, Mathematical Intelligencer 6(3), 1984

ICS 35.240.70

Price based on 65 pages