INTERNATIONAL STANDARD

ISO 19108

First edition
2002-09-01

Geographic information — Temporal schema

Information géographique — Schéma temporel

Reference number
ISO 19108:2002(E)

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.ch
Web www.iso.ch

Printed 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 3.

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 International Standard may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 19108 was prepared by Technical Committee ISO/TC 211, *Geographic information/Geomatics.*

Annexes A and C form a normative part of this International Standard. Annexes B and D are for information only.

Introduction

This International Standard defines the standard concepts needed to describe the temporal characteristics of geographic information as they are abstracted from the real world. Temporal characteristics of geographic information include feature attributes, feature operations, feature associations, and metadata elements that take a value in the temporal domain.

The widespread application of computers and geographic information systems has led to the increased analysis of geospatial data within multiple disciplines. Geographic information is not confined to a three-dimensional spatial domain. Many geographic information systems require data with temporal characteristics. A standardized conceptual schema for temporal characteristics will increase the ability of geographic information to be used for certain types of applications such as simulations and predictive modelling.

As a fundamental physical reality, time is of interest to the whole range of scientific and technical disciplines. Many of the concepts described in this International Standard are applicable outside of the field of geographic information. ISO/TC 211 does not intend to develop independent standards for the description of time, but the technical committee believes that it is necessary to standardize the way to describe the temporal characteristics of geographic data sets and features. Geographic information system and software developers and users of geographic information will use this schema to provide consistently understandable temporal data structures.

Historically, temporal characteristics of features have been treated as thematic feature attributes. For example, a feature "Building" may have an attribute "date of construction". However, there is increasing interest in describing the behaviour of features as a function of time. This can be supported to a limited extent when time is treated independently of space. For example, the path followed by a moving object can be represented as a set of features called "way point", each of which is represented as a point and has an attribute that provides the time at which the object was at that spatial position. Behaviour in time may be described more easily if the temporal dimension is combined with the spatial dimensions, so that a feature can be represented as a spatiotemporal object. For example, the path of a moving object could be represented as a curve described by coordinates in x, y and t. This International Standard has been prepared in order to standardize the use of time in feature attributes. Although it does not describe feature geometry in terms of a combination of spatial and temporal coordinates, it has been written to establish a basis for doing so in a future standard within the ISO 19100 series.

Geographic information — Temporal schema

1 Scope

This International Standard defines concepts for describing temporal characteristics of geographic information. It depends upon existing information technology standards for the interchange of temporal information. It provides a basis for defining temporal feature attributes, feature operations, and feature associations, and for defining the temporal aspects of metadata about geographic information. Since this International Standard is concerned with the temporal characteristics of geographic information as they are abstracted from the real world, it emphasizes valid time rather than transaction time.

2 Conformance

2.1 Conformance classes and requirements

This International Standard defines five conformance classes, which depend upon the nature of the test item.

2.2 Application schemas for data transfer

To conform to this International Standard, an application schema for data transfer shall satisfy the requirements of A.1 of the Abstract Test Suite in annex A.

2.3 Application schemas for data with operations

To conform to this International Standard, an application schema that supports operations on data shall satisfy the requirements of A.2 of the Abstract Test Suite in annex A.

2.4 Feature catalogues

To conform to this International Standard, a feature catalogue shall satisfy the requirements of A.3 of the Abstract Test Suite in annex A.

2.5 Metadata element specifications

To conform to this International Standard, a metadata specification shall satisfy the requirements of A.4 of the Abstract Test Suite in annex A.

2.6 Metadata for data sets

To conform to this International Standard, metadata for a data set shall satisfy the requirements of A.5 of the Abstract Test Suite in annex A.

3 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this International Standard. For dated references, subsequent amendments to, or revisions of, any of these

publications do not apply. However, parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO 31-1:1992, *Quantities and units — Part 1: Space and time*

ISO 1000:1992, *SI units and recommendations for the use of their multiples and of certain other units*

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

ISO/IEC 11404:1996, *Information technology — Programming languages, their environments and system software interfaces — Language-independent data types*

ISO/TS 19103:—[1)], *Geographic information — Conceptual schema language*

ISO 19107:—[1)], *Geographic information — Spatial schema*

ISO 19109:—[1)], *Geographic information — Rules for application schema*

ISO 19110:—[1)], *Geographic information — Methodology for feature cataloguing*

ISO 19111:—[1)], *Geographic information — Spatial referencing by coordinates*

ISO 19115:—[1)], *Geographic information — Metadata*

4 Terms, definitions and abbreviated terms

4.1 Terms and definitions

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

4.1.1
calendar
discrete **temporal reference system** that provides a basis for defining **temporal position** to a resolution of one **day**

4.1.2
calendar era
sequence of **periods** of one of the types used in a **calendar**, counted from a specified **event**

4.1.3
UTC
Coordinated Universal Time
time scale maintained by the Bureau International des Poids et Mesures (International Bureau of Weights and Measures) and the International Earth Rotation Service (IERS) that forms the basis of a coordinated dissemination of standard frequencies and time signals [ITU-R Rec.TF.686-1 (1997)]

4.1.4
day
period having a **duration** nominally equivalent to the **periodic time** of the Earth's rotation around its axis

---

1) To be published.

4.1.5
edge
one-dimensional **topological primitive** [ISO 19107]

NOTE The geometric realization of an edge is a curve. The boundary of an edge is the set of one or two nodes associated to the edge within a topological complex.

4.1.6
event
action which occurs at an **instant**

4.1.7
feature
abstraction of real world phenomena [ISO 19101]

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

4.1.8
feature association
relationship between features [ISO 19109]

NOTE 1 A feature association may occur as a type or an instance. Feature association type or feature association instance is used when only one is meant.

NOTE 2 Feature associations include aggregation of features.

4.1.9
feature attribute
characteristic of a **feature** [Adapted from ISO 19110]

NOTE A feature attribute has a name, a data type, and a value domain associated to it.

4.1.10
feature division
feature succession in which a previously existing **feature** is replaced by two or more distinct **feature** instances of the same **feature** type

EXAMPLE An instance of the feature type “land parcel” is replaced by two instances of the same type when the parcel is legally subdivided.

4.1.11
feature fusion
feature succession in which two or more previously existing instances of a **feature** type are replaced by a single instance of the same **feature** type

EXAMPLE Two instances of the feature type “pasture” are replaced by a single instance when the fence between the pastures is removed.

4.1.12
feature operation
operation that every instance of a **feature** type may perform [ISO 19110]

EXAMPLE An operation upon a “dam” is to raise the dam. The results of this operation are to raise the height of the “dam” and the level of water in a “reservoir”.

NOTE Feature operations provide a basis for feature type definition.

4.1.13
feature substitution
feature succession in which one **feature** instance is replaced by another **feature** instance of the same or different **feature** type

EXAMPLE An instance of feature type “building” is razed and replaced by an instance of feature type “parking lot”.

4.1.14
feature succession
replacement of one or more **feature** instances by other **feature** instances, such that the first **feature** instances cease to exist

4.1.15
geometric primitive
object representing a single, connected, homogeneous element of space [ISO 19107]

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

4.1.16
Gregorian calendar
calendar in general use; first introduced in 1582 to define a year that more closely approximated the tropical year than the Julian **calendar** [adapted from ISO 8601:2000]

NOTE 1 The introduction of the Gregorian calendar included the cancellation of the accumulated inaccuracies of the Julian year. In the Gregorian calendar, a calendar year is either a common year or a leap year; each year is divided into 12 sequential months.

4.1.17
instant
0-dimensional **geometric primitive** representing position in time

NOTE The geometry of time is discussed in 5.2.

4.1.18
interval scale
scale with an arbitrary origin which can be used to describe both ordering of values and distances between values

NOTE Ratios of values measured on an interval scale have no meaning.

4.1.19
Julian date
Julian day number followed by the decimal fraction of the **day** elapsed since the preceding noon

4.1.20
Julian day number
number of **days** elapsed since Greenwich mean noon on 1 January 4713 BC, Julian proleptic calendar

4.1.21
life span
period during which something exists

NOTE Valid-time life span is the period during which an object exists in the modelled reality. Transaction-time life span is the period during which a database object is current in the database.

4.1.22
month
period approximately equal in **duration** to the **periodic time** of a lunar **cycle**

NOTE The duration of a month is an integer number of days. The number of days in a month is determined by the rules of the particular calendar.

4.1.23
node
0-dimensional **topological primitive** [ISO 19107]

NOTE The boundary of a node is the empty set.

4.1.24
ordinal era
one of a set of named **periods** ordered in time

4.1.25
ordinal scale
scale which provides a basis for measuring only the relative position of an object

4.1.26
ordinal temporal reference system
temporal reference system composed of **ordinal eras**

4.1.27
period
one-dimensional **geometric primitive** representing extent in time

NOTE A period is bounded by two different temporal positions.

4.1.28
periodic time
duration of one **cycle** [adapted from ISO 31-2:1992]

4.1.29
point
0-dimensional **geometric primitive**, representing a position [ISO 19107]

NOTE The boundary of a point is the empty set.

4.1.30
temporal coordinate
distance from the origin of the **interval scale** used as the basis for a **temporal coordinate system**

4.1.31
temporal coordinate system
temporal reference system based on an **interval scale** on which distance is measured as a multiple of a single unit of time

4.1.32
temporal feature association
feature association characterized by a reference to time or to a temporal constraint

4.1.33
temporal feature operation
feature operation specified as a function of time

4.1.34
temporal position
location relative to a **temporal reference system**

4.1.35
temporal reference system
reference system against which time is measured

4.1.36
topological complex
collection of **topological primitives** that is closed under the boundary operations [ISO 19107]

NOTE Closed under the boundary operations means that if a topological primitive is in the topological complex, then its boundary objects are also in the topological complex.

4.1.37
topological primitive
topological object that represents a single, non-decomposable element [ISO 19107]

NOTE A topological primitive corresponds to the interior of a geometric primitive of the same dimension in a geometric realization.

4.1.38
transaction time
time when a fact is current in a database and may be retrieved [Jensen et al. (1994)]

4.1.39
valid time
time when a fact is true in the abstracted reality [Jensen et al. (1994)]

4.2 Abbreviated terms

For the purposes of this International Standard, the following abbreviations apply.

AD	Anno Domini
BC	Before Christ
GPS	Global Positioning System
TOW	Time of Week
UML	Unified Modeling Language
UTC	Coordinated Universal Time
WN	Week Number

5 Conceptual schema for temporal aspects of geographic information

5.1 Structure of the schema

This clause presents a conceptual schema for describing temporal aspects of geographic information. The schema is specified in the Unified Modeling Language (UML) [Object Management Group (1999)]. ISO/TS 19103 describes the way in which UML is used in this family of standards. The three primary aspects of a UML class are attributes, operations, and associations. This schema uses all three. This schema is an abstract model; to conform to this International Standard, an implementation shall provide the capabilities described by these elements of the abstract model, but it need not implement them in the same way.

The schema consists of two packages (see Figure 1). The package Temporal Objects (described in 5.2) defines temporal geometric and topological objects that shall be used as values for the temporal characteristics of features and data sets. The temporal position of an object shall be specified in relation to a temporal reference system. The package Temporal Reference System (5.3, 5.4) provides elements for describing temporal reference systems. Subclause 5.5 describes how the concepts specified in 5.2 through 5.4 shall be used in the context of geographic information.

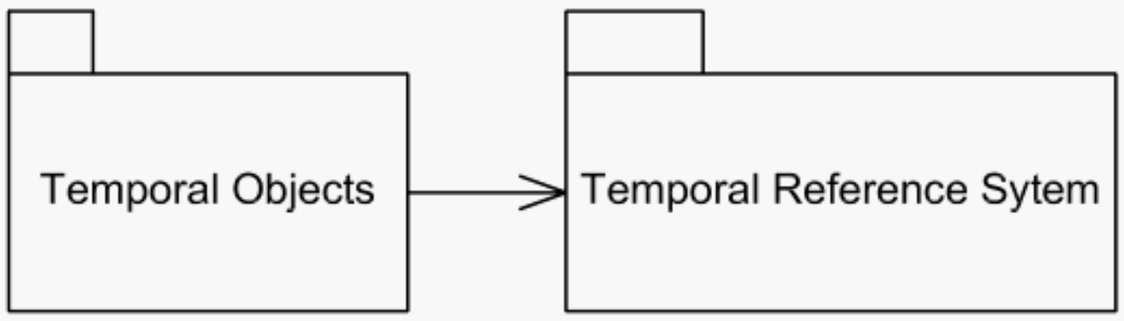

Figure 1 — Structure of the temporal schema

Names of UML classes defined in the ISO 19100 series of standards begin with a two-letter prefix followed by an underscore to identify the specific standard, and possibly the package, in which they are defined. TM_ is used to identify classes defined in this International Standard.

5.2 Geometry of time

5.2.1 Time as a dimension

Time is a dimension analogous to any of the spatial dimensions. Like space, time has geometry and topology. A point in time occupies a position that can be identified in relation to a temporal reference system. Distance can be measured. Unlike space, however, time has a single dimension — temporal reference systems are analogous to the linear referencing systems that are used to describe spatial position for some kinds of applications. Although time has an absolute directionality — movement in time is always forward — time can be measured in two directions.

NOTE Although time always has geometry and topology at a conceptual level, sometimes it is possible or desirable to describe geometry alone, or topology alone.

Time is measured on two types of scales, ordinal and interval. An ordinal scale provides information only about relative position in time, while an interval scale offers a basis for measuring duration.

5.2.2 Temporal objects

Temporal geometric and topological objects shall be used as values for the temporal characteristics of features and data sets. See 5.5 and annex B for an explanation and examples. TM_Object (see Figure 2) is an abstract class that has two subclasses. TM_Primitive is an abstract class that represents a non-decomposed element of geometry or topology of time. There are two subclasses of TM_Primitive. A TM_GeometricPrimitive (5.2.3) provides information about temporal position. A TM_TopologicalPrimitive (5.2.4.2) provides information about connectivity in time. A TM_Complex is an aggregation of TM_Primitives. TM_TopologicalComplex (5.2.4.5) is the only subclass of TM_Complex that is defined in this International Standard; it is an aggregation of connected TM_TopologicalPrimitives.

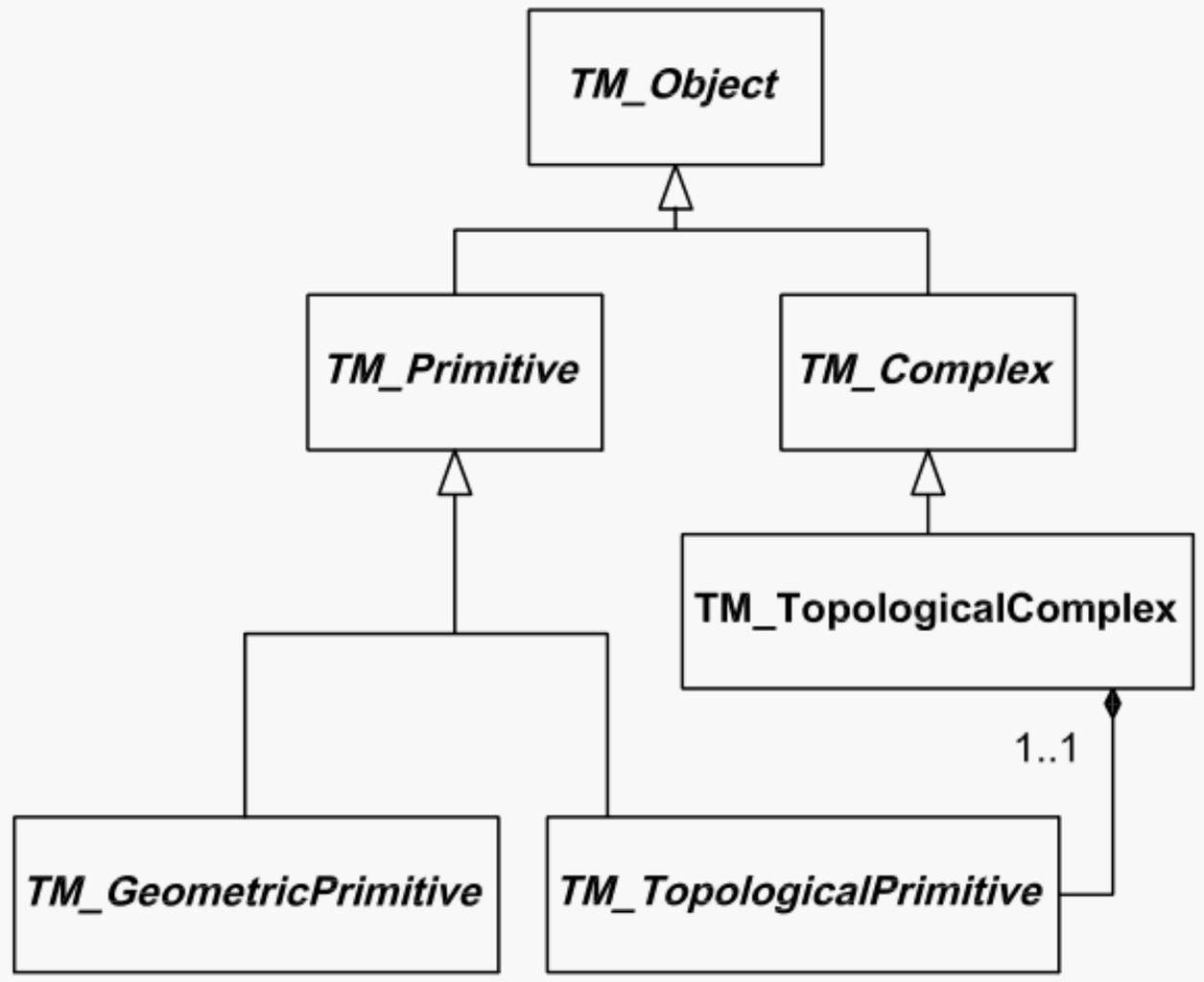

Figure 2 — Temporal objects

5.2.3 Temporal geometric primitives

5.2.3.1 Temporal geometric primitive classes

The two geometric primitives in the temporal dimension are the instant and the period. These primitives are defined analytically in the case of time measured on an interval scale, and analogically in the case of time measured on an ordinal scale. TM_GeometricPrimitive is an abstract class with two subclasses, TM_Instant represents an instant and TM_Period represents a period (see Figure 3). TM_GeometricPrimitive inherits from TM_Primitive a dependency on the interface TM_Order, and also has a dependency on the interface TM_Separation. The «uses» stereotype on the dependency means that the class may support any of the operations defined for the interface, but need not support all of them.

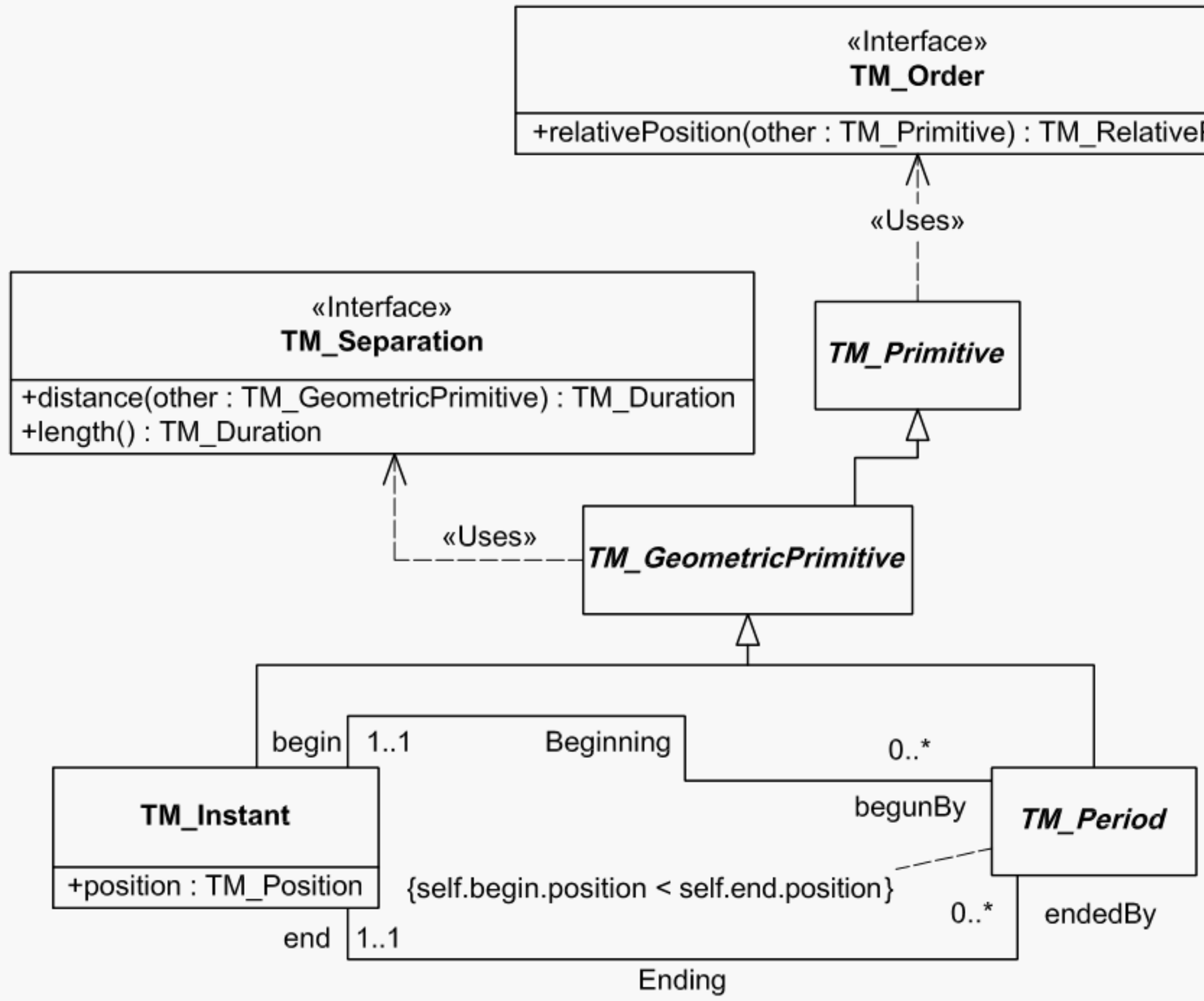

Figure 3 — Temporal geometric primitives

5.2.3.2 TM_Instant

An instant is a zero-dimensional geometric primitive that represents position in time. It is equivalent to a point in space. In practice, an instant is an interval whose duration is less than the resolution of the time scale.

Attributes:

TM_Instant has one attribute.

a) *position: TM_TemporalPosition* shall provide the position of this TM_Instant. The TM_TemporalPosition shall be associated with a single temporal reference system, as specified in 5.3. An instance of TM_Instant is an identifiable object, while an instance of TM_TemporalPosition is a data value. The TM_TemporalPosition of a given TM_Instant may be replaced by an equivalent TM_TemporalPosition associated with a different temporal reference system.

5.2.3.3 TM_Period

The period is a one-dimensional geometric primitive that represents extent in time. The period is equivalent to a curve in space. Like a curve, it is an open interval bounded by beginning and end points (instants), and has length (duration). Its location in time is described by the temporal positions of the instants at which it begins and ends; its duration equals the temporal distance between those two temporal positions.

Since it is impossible to measure duration on an ordinal scale, an instant cannot be distinguished from a period on this basis. In practice, the time at which a single event occurs can be considered an instant when time is measured

on an ordinal scale. A series of consecutive events must occupy an interval of time, which is a period. The term period is commonly applied to sequences of events that have distinctive characteristics in common.

Associations:

a) *Beginning* links the TM_Period to the TM_Instant at which it starts.

b) *Ending* links the TM_Period to the TM_Instant at which it ends.

For a variety of reasons, the position of the TM_Instant designated by *begin* or *end* may be indeterminate. See 5.4.3 for a discussion of indeterminate temporal positions.

Constraints:

a) {self.begin.position < self.end.position} states that the temporal position of the beginning of the period must be less than (i.e. earlier than) the temporal position of the end of the period.

5.2.3.4 TM_Order

TM_GeometricPrimitives inherit a dependency on TM_Order from TM_Primitive. TM_Order provides an operation for determining the position of this TM_Primitive relative to another TM_Primitive.

Operation:

a) *relativePosition (other:TM_Primitive): TM_RelativePosition* shall accept another TM_Primitive as input and return a value for TM_RelativePosition as specified in 5.2.3.5.

5.2.3.5 TM_RelativePosition

Values for relative positions are provided by the enumerated data type TM_RelativePosition (see Figure 4) and are based on the 13 temporal relationships identified by Allen (1983). For TM_Primitives, the operation *TM_Order.relativePosition* shall return a value of the TM_RelativePosition as follows:

a) If both this TM_Primitive and other are TM_Instants, the operation shall return a value for TM_RelativePosition as follows:

Returns:	If:
Before	self.position<other.position
Equals	self.position=other.position
After	self.position>other.position

b) If this TM_Primitive is a TM_Period and other is a TM_Instant, the operation shall return a value for TM_RelativePosition as follows:

Returns:	If:
Before	self.end.position < other.position
EndedBy	self.end.position = other.position
Contains	self.begin.position < other.position AND self.end > other.position
BegunBy	self.begin.position = other.position
After	self.begin.position > other.position

c) If this TM_Primitive is a TM_Instant and other is a TM_Period, the operation shall return a value for TM_RelativePosition as follows:

Returns:	If:
Before	self.position < other.begin.position
Begins	self.position = other.begin.position
During	self.position > other.begin.position AND self.position < other.end.position
Ends	self.position = other.end.position
After	self.position > other.end.position

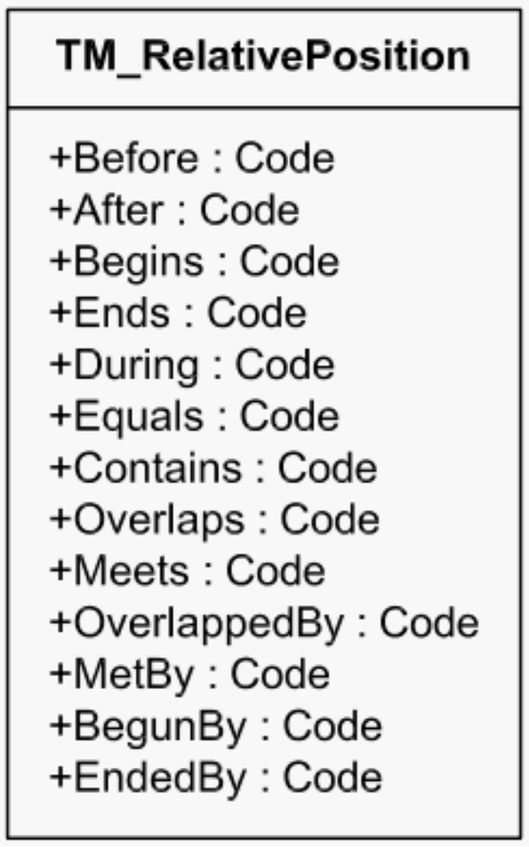

Figure 4 — TM_RelativePosition

d) If both this TM_Primitive and other are TM_Periods, the operation shall return a value for TM_RelativePosition as follows:

Returns:	If:
Before	self.end.position < other.begin.position
Meets	self.end.position = other.begin.position
Overlaps	self.begin.position < other.begin.position AND self.end.position > other.begin.position AND self.end.position < other.end.position
Begins	self.begin.position = other.begin.position AND self.end.position < other.end.position
BegunBy	self.begin.position = other.begin.position AND self.end.position > other.end.position
During	self.begin.position > other.begin.position AND self.end.position < other.end.position
Contains	self.begin.position < other.begin.position AND self.end.position > other.end.position
Equals	self.begin.position = other.begin.position AND self.end = other.end.position
OverlappedBy	self.begin.position > other.begin.position AND self.begin.position < other.end.position AND self.end.position > other.end.position
Ends	self.begin.position > other.begin.position AND self.end.position = other.end.position
EndedBy	self.begin.position < other.begin.position AND self.end.position = other.end.position
MetBy	self.begin.position = other.end.position
After	self.begin.position > other.end.position

This operation shall raise an exception if any input value of TM_TemporalPosition is indeterminate.

5.2.3.6 TM_Separation

TM_GeometricPrimitive also has a dependency on the interface TM_Separation, which provides operations for calculating length and distance. TM_Duration (see Figure 5) is a data type that contains return values for those operations.

a) *length(): TM_Duration* shall return the duration of this TM_GeometricPrimitive. The length of a TM_Instant is zero by definition. When the TM_GeometricPrimitive is a TM_Period, the operation shall return the distance between the temporal positions provided by TM_Period.begin and TM_Period.end. This operation shall raise an exception if the value of either TM_TemporalPosition is indeterminate, or if the TM_TemporalPositions are referenced to a TM_OrdinalReferenceSystem.

b) *distance (other:TM_GeometricPrimitive): TM_Duration* shall return the distance from this TM_GeometricPrimitive to another TM_GeometricPrimitive, i.e. the absolute value of the difference between their temporal positions. The distance shall be the distance between the two closest TM_TemporalPositions of the two TM_GeometricPrimitives. If either TM_GeometricPrimitive is connected to, overlaps, or is contained within the other, the operation shall return a value of zero. The operation shall raise an exception if: (1) either of the two TM_TemporalPositions is indeterminate, (2) the TM_TemporalPositions are not both associated with the same TM_ReferenceSystem, or (3) either TM_TemporalPosition is associated with a TM_OrdinalReferenceSystem.

5.2.3.7 TM_Duration

TM_Duration (see Figure 5) is a data type to be used for describing length or distance in the temporal dimension. It has two subtypes.

TM_PeriodDuration uses the format specified by ISO 8601 for exchanging information about the duration of a period. It allows duration to be expressed in terms of multiple units of time, specifically years, months, days, hours, minutes, and seconds. Although individual values are optional, a value shall be provided for at least one unit.

Attributes:

a) *designator: CharacterString = P* is a mandatory element which designates that the following characters represent the duration of a period.

b) *years [0..1]: CharacterString* is a positive integer, followed by the character “Y” which indicates the number of years in the period.

c) *months [0..1]: CharacterString* is a positive integer followed by the character “M” which indicates the number of months in the period.

d) *days [0..1]: CharacterString* is a positive integer followed by the character “D” which indicates the number of days in the period.

e) *timeIndicator [0..1]: CharacterString* = “T” shall be included whenever the sequence includes values for units of less than a day.

f) *hours [0..1]: CharacterString* is a positive integer followed by the character “H” which indicates the number of hours in the period.

g) *minutes [0..1]: CharacterString* is a positive integer followed by the character “M” which indicates the number of minutes in the period.

h) *seconds [0..1]: CharacterString* is a positive integer followed by the character “S” which indicates the number of seconds in the period.

The value for the rightmost unit may be expressed as a positive decimal fraction rather than as a positive integer.

EXAMPLE A duration of five days, four hours, and 30,7 minutes is represented as P5DT4H30.7M.

NOTE Although this format is defined in ISO 8601 for use with dates in the Gregorian calendar and times in UTC, TM_PeriodDuration may be used as the data type for expressing length or distance whenever temporal positions are referenced to a calendar that describes dates in terms of years, months and days and a clock that describes times in hours, minutes and seconds.

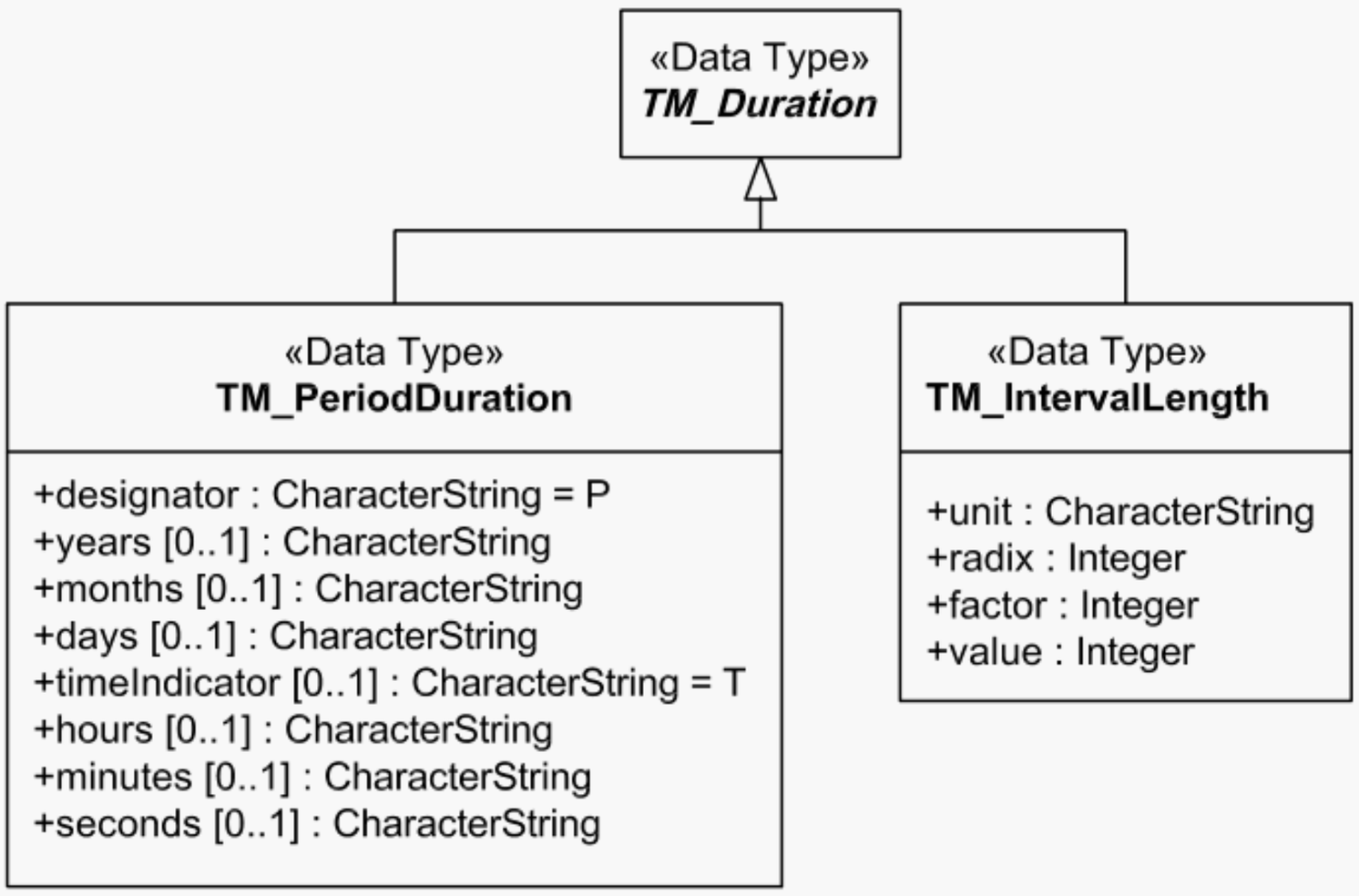

Figure 5 — TM_Duration

TM_IntervalLength is a data type specified by ISO/IEC 11404 for intervals of time, represented here in UML. It supports the expression of duration in terms of a specified multiple of a single unit of time.

Attributes:

a) *unit: CharacterString* is the name of the unit of measure used to express the length of the interval.

b) *radix: Integer* is a positive integer that is the base of the multiplier of the unit.

c) *factor: Integer* is an integer that is the exponent of the base.

d) *value: Integer* is the length of the time interval as an integer multiple of one radix $^{(-factor)}$ of the specified unit.

EXAMPLE Unit = “second,” radix = 10, factor = 3, value = 7 specifies a time interval length of 7 ms.

5.2.4 Temporal topological objects

5.2.4.1 Introduction

Topology provides information about connectivity between objects in time, and may incidentally provide information about the ordering of objects in time. It does not provide information about temporal position. Topological relationships can often be derived from geometric information. However, data about temporal position is sometimes inadequate for doing this, so topology may need to be described explicitly. Topology may also be used in applications that have a requirement to describe topological relationships explicitly, even though they could be derived.

EXAMPLE It may be possible to observe the order of several events or states within a single ordinal era, but the ordinal temporal reference system does not support assignment of distinct temporal positions to these events or states. The order can be described by modelling these events or states with topological primitives.

5.2.4.2 TM_TopologicalPrimitive

A topological primitive represents a single non-decomposable element of topology and its relationships to other topological primitives within a topological complex. The two topological primitives relevant for temporal information are the node, which is 0-dimensional, and the edge, which is one-dimensional. In the temporal schema, there are presented by two subclasses of TM_TopologicalPrimitive: TM_Node and TM_Edge (see Figure 6). When an application includes information about temporal position as well as connectivity, a TM_TopologicalPrimitive may be associated with a TM_GeometricPrimitive of the same dimension. Because topological primitives are intended to provide information about connectivity, their most significant characteristics are the associations that link them to each other. Another consequence is the requirement that every TM_TopologicalPrimitive shall be a member of one and only one TM_TopologicalComplex.

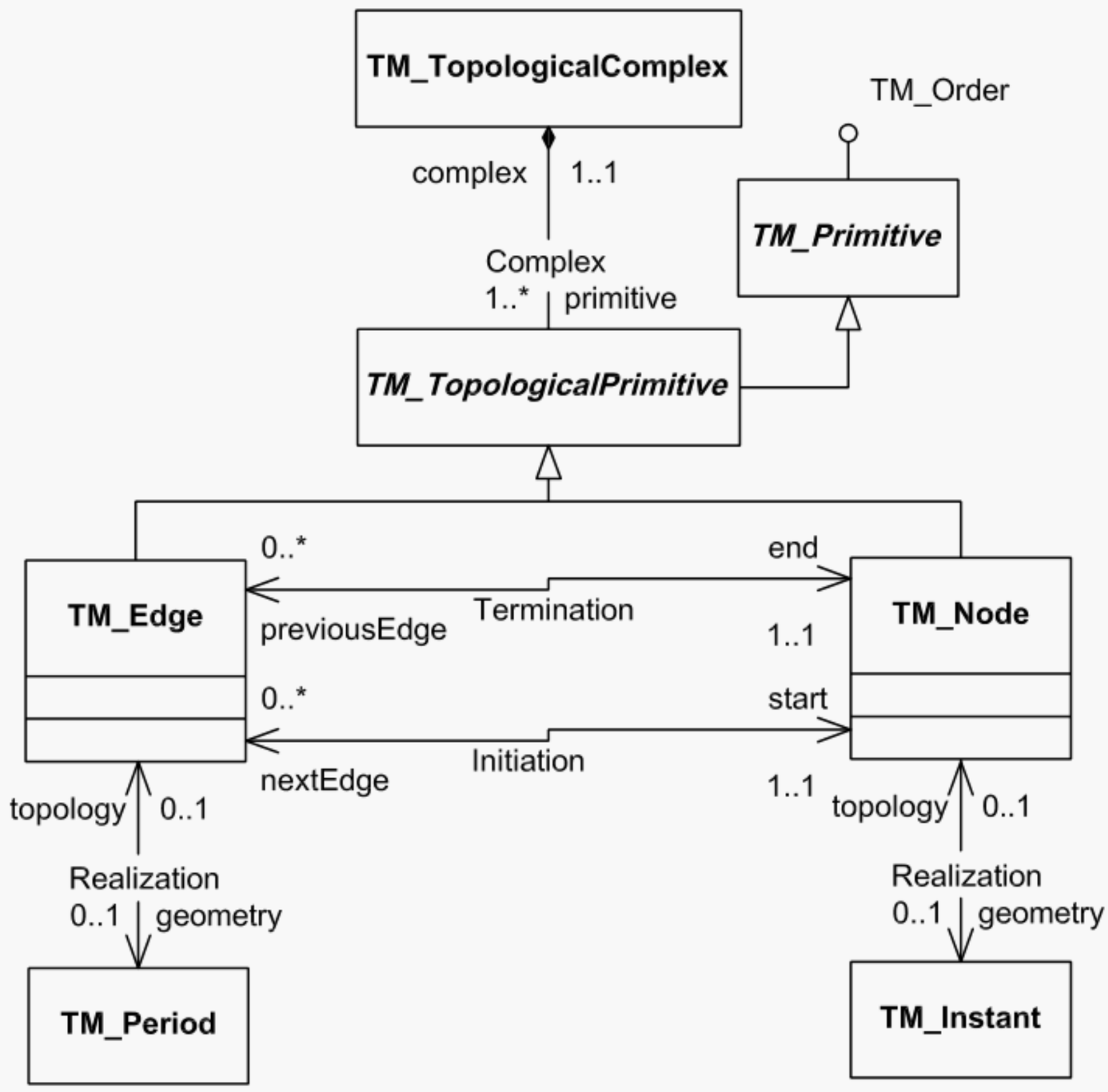

Figure 6 — Topology of time

5.2.4.3 TM_Node

The TM_Node is the zero dimensional topological primitive in time. Its geometric realization is a TM_Instant.

Associations:

TM_Node shall support three associations:

a) *Initiation* shall link this TM_Node to the TM_Edges for which it is the start.

b) *Termination* shall link this TM_Node to the TM_Edges for which it is the end.

c) *Realization* is an optional association that may link this TM_Node to its corresponding TM_Instant. Only one TM_Node may be associated with a TM_Instant, and only one TM_Instant may be associated with a TM_Node.

5.2.4.4 TM_Edge

The TM_Edge is the one-dimensional topological primitive in time. It corresponds to a TM_Period.

Associations:

TM_Edge shall support three associations:

a) *Initiation* shall link this TM_Edge to the TM_Node that is its start. A TM_Edge may have one and only one start node.

b) *Termination* shall link this TM_Edge to the TM_Node that is its end. A TM_Edge may have one and only one end node.

c) *Realization* is an optional association that shall link this TM_Edge to its corresponding TM_Period. Only one TM_Edge may be associated with a TM_Period, and only one TM_Period may be associated with a TM_Edge.

5.2.4.5 TM_TopologicalComplex

A topological complex is a set of connected topological primitives. A topological primitive is always connected to one or more other topological primitives, and is, therefore, always a member of a topological complex. Temporal topological complexes are represented in this schema by the class TM_TopologicalComplex.

Associations:

a) *Composition* shall link the TM_TopologicalComplex to the set of TM_TopologicalPrimitives that it includes. Since each TM_Edge in the TM_TopologicalComplex is linked to two TM_Nodes, the minimum number of TM_Nodes in the complex is two.

5.2.4.6 Linear and non-linear graphs

5.2.4.6.1 Non-linear graph

The multiplicities at the TM_Edge end of the associations *Initiation* and *Termination* (see Figure 6) allow non-linear topology. A TM_Node may be the *startNode*, or the *endNode*, for more than one TM_Edge. The TM_Edges that share a *startNode* or an *endNode* are assumed to be separated in some way, either because they represent temporal characteristics of different features, or because they represent different temporal characteristics of the same feature.

NOTE Non-linear topology of time is analogous to the case of non-planar topology in space. In both cases, it is assumed that topological primitives that seem to intersect or overlap are actually separated in an unmeasured additional dimension.

5.2.4.6.2 Linear graph

Because time is a single dimension, temporal topology should be represented as a linear graph. In linear topology, a TM_TopologicalComplex is a sequence of TM_Primitives in which TM_Nodes alternate with TM_Edges. The first element of the sequence is the *startNode* of the first TM_Edge in the sequence, and the last element is the *endNode* of the last TM_Edge in the sequence. To restrict an application schema to linear topology, the multiplicities at the TM_Edge end of the associations *Initiation* and *Termination* shall be restricted to 0..1, so that every TM_Node other than the first and the last shall be connected to two, and only two, TM_Edges, a previousEdge and a nextEdge.

5.2.4.7 TM_Order

TM_TopologicalPrimitives inherits the interface TM_Order from TM_Primitive. TM_Order provides an operation for determining the position of this TM_Primitive relative to another TM_Primitive.

Operation

a) *RelativePosition (other:TM_Primitive): TM_RelativePosition* shall accept a TMPrimitive as input and return a TM_RelativePosition as specified below.

The relative positions of two TM_TopologicalPrimitives depend upon the positions they occupy within the sequence of TM_TopologicalPrimitives that makes up a TM_TopologicalComplex. For TM_TopologicalPrimitives, this operation shall return a value of the enumerated data type TM_RelativePosition (see Figure 4), as follows:

Returns:	If:
Before	This TM_TopologicalPrimitive is earlier in the sequence than *other* and is not linked to *other* in an *Initiation* or *Termination* association.
Meets	The two TM_TopologicalPrimitives are TM_Edges associated to the same TM_Node, where this TM_Edge is linked to the TM_Node as a *previousEdge* in a *Termination* association, and *other* is linked to the TM_Node as a *nextEdge* in an *Initiation* association.
Begins	This TM_TopologicalPrimitive is TM_Node, *other* is a TM_Edge, and these two TM_Primitives are linked in an *Initiation* association.
BegunBy	This TM_TopologicalPrimitive is TM_Edge, *other* is a TM_Node, and these two TM_Primitives are linked in an *Initiation* association.
Equals	This TM_TopologicalPrimitive and *other* are the same.
Ends	This TM_TopologicalPrimitive is TM_Node, other is a TM_Edge, and these two TM_Primitives are linked in a *Termination* association.
EndedBy	This TM_TopologicalPrimitive is TM_Edge, *other* is a TM_Node, and these two TM_Primitives are linked in a *Termination* association.
MetBy	The two TM_TopologicalPrimitives are TM_Edges associated to the same TM_Node; where this TM_Edge is linked to the TM_Node as a nextEdge in an *Initiation* association, and *other* is linked to the TM_Node as a *previousEdge* in a *Termination* association.
After	This TM_TopologicalPrimitive is later in the sequence than *other* and is not linked to *other* in an *Initiation* or *Termination* association.

The operation shall raise an exception if the two TM_TopologicalPrimitives are not in the same TM_TopologicalComplex.

5.3 Temporal reference systems

5.3.1 Types of temporal reference systems

A value in the time domain is a temporal position measured relative to a temporal reference system. ISO 8601 specifies the use of the Gregorian Calendar and 24-hour local or Coordinated Universal Time (UTC) for information interchange. This shall be the primary temporal reference system for use with geographic information. A different temporal reference system may be appropriate for some applications of geographic information. In this case, the feature catalogue or the metadata associated with an application schema or data set shall include either a citation for a document that describes that temporal reference system, or a description of that system. When more than one temporal reference system is used in a single feature catalogue, application schema, or data set, the definition of each temporal characteristic shall identify the temporal reference system that is used. This subclause describes a conceptual schema that shall be used as the basis for such descriptions. Annex C defines metadata elements derived from this schema that shall be used in such a description.

The Temporal reference system package includes three common types of temporal reference systems: calendars (used with clocks for greater resolution), temporal coordinate systems, and ordinal temporal reference systems (see Figure 7).

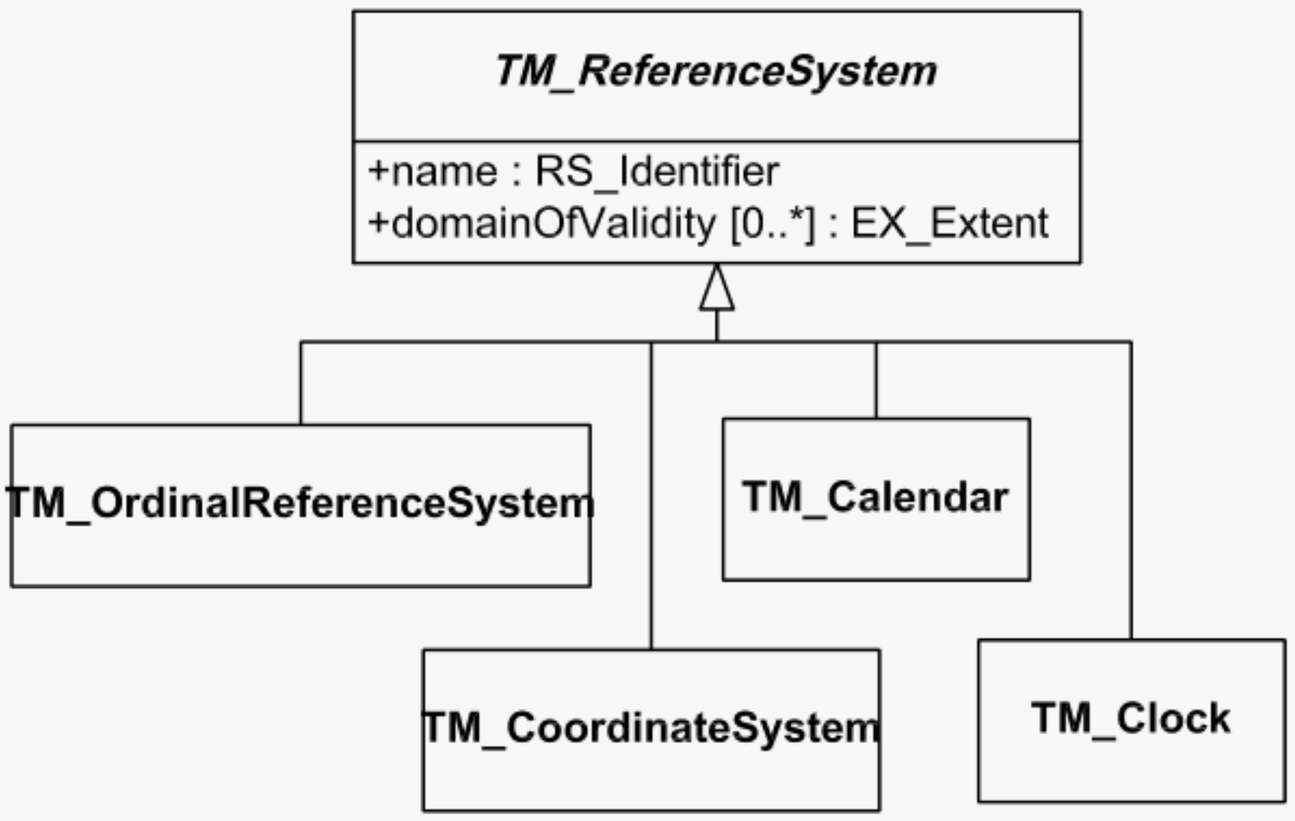

Figure 7 — Temporal reference systems

The class TM_ReferenceSystem shall provide the following attributes.

a) *name: RS_Identifier* shall provide a name that uniquely identifies the temporal reference system. The data type RS_Identifier is defined in ISO 19111.

b) *DomainOfValidity: EX_Extent* shall identify the space and time within which the TM_ReferenceSystem is applicable. The data type EX_Extent is specified in ISO/TS 19103. It permits a description of both spatial and temporal extent. This attribute shall be used whenever an application schema includes TM_TemporalPositions referenced to a TM_ReferenceSystem which has a valid extent that is less than the extent of a data set containing such values.

The following three subclauses describe the schemas for the three reference system types.

5.3.2 Calendars and clocks

5.3.2.1 Introduction

Calendars and clocks are both based on interval scales. A calendar is a discrete temporal reference system that provides a basis for defining temporal position to a resolution of one day. A clock provides a basis for defining temporal position within a day. A clock must be used with a calendar in order to provide a complete description of a temporal position within a specific day. Figure 8 provides the details of the classes TM_Calendar and TM_Clock.

Calendars have a variety of complex internal structures. This schema defines a simple external calendar interface. Annex D provides a more detailed description of the internal structure of calendars.

Every calendar provides a set of rules for composing a calendar date from a set of elements such as year, month, and day. In every calendar, years are numbered relative to the date of a reference event that defines a calendar era. A single calendar may reference more than one calendar era (See D.3.1 and D.3.2 for examples).

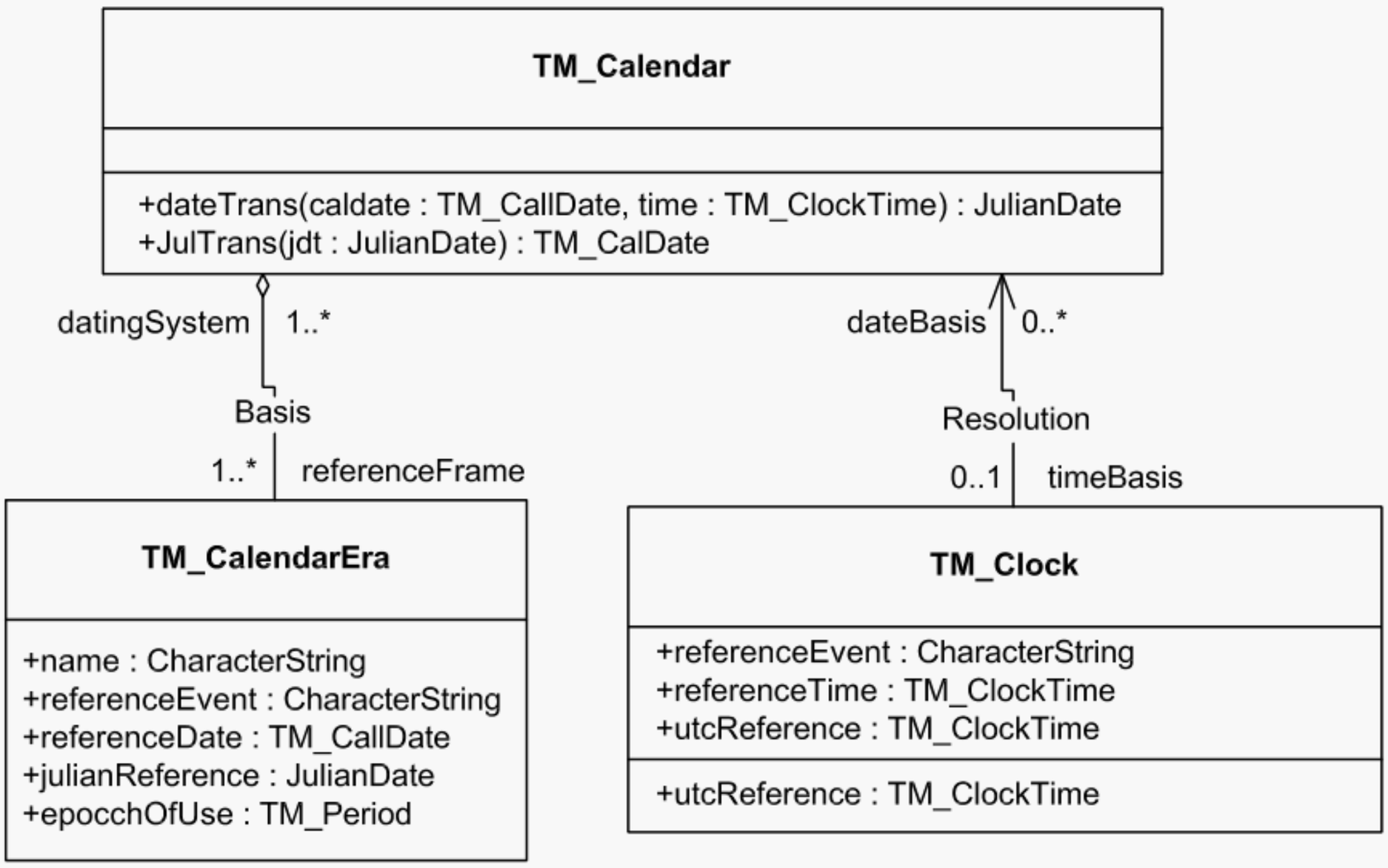

Figure 8 — Calendar and clock

5.3.2.2 Calendar era

The class TM_CalendarEra shall contain the following attributes:

a) *name: CharacterString* shall uniquely identify the calendar era within this calendar.

b) *referenceEvent: CharacterString* shall provide the name or description of a mythical or historic event which fixes the position of the base scale of the calendar era.

c) *referenceDate: TM_CalDate* shall provide the date of the referenceEvent expressed as a date in the given calendar. In most calendars, this date is the origin (i.e. the first day) of the scale, but this is not always true.

d) *julianReference: JulianDate* shall provide the Julian date that corresponds to the reference date.

e) *epochOfUse: TM_Period* shall identify the TM_Period for which the calendar era was used as a basis for dating. The data type for *TM_Period.begin* and *TM_Period.end* shall be JulianDate (5.4.5.2).

Associations:

a) *Basis* shall link this TM_CalendarEra to those TM_Calendars that use this TM_CalendarEra as a reference for dating.

5.3.2.3 Calendar

TM_Calendar shall support the following operations:

a) *dateTrans (calDate: TM_CalDate, time: TM_ClockTime): JulianDate* shall accept a date in the specified calendar and a time in the specified clock as input and return a Julian date.

b) *julTrans (jdt: JulianDate): TM_CalDate* shall accept a Julian date as input and return a date in this calendar.

NOTE The Julian day numbering system is a temporal coordinate system that has an origin earlier than any known calendar. Transforming calendar dates to and from Julian dates provides a relatively simple basis for transforming dates from one calendar to another.

Any description of the internal structure of a specific calendar shall include sufficient information to enable a user to implement these operations. It shall include a description of each calendar era associated with the calendar, and it shall provide sufficient information to enable mapping of a date in that calendar to the equivalent Julian date.

Associations:

a) *Basis* shall link this TM_Calendar to the TM_CalendarEras that it uses as a reference for dating.

b) *Resolution* shall link this TM_Calendar to the TM_Clock that is used for specifying temporal positions within the smallest calendar interval.

5.3.2.4 Clock

TM_Clock shall contain the following attributes:

a) *referenceEvent: CharacterString* shall provide the name or description of an event, such as solar noon or sunrise, which fixes the position of the base scale of the clock.

b) *referenceTime: TM_ClockTime* shall provide the time of day associated with the reference event expressed as a time of day in the given clock. The reference time is usually the origin of the clock scale.

c) *utcReference: TM_ClockTime* shall provide the 24-hour local or UTC time that corresponds to the reference time.

TM_Clock shall support the following operations:

a) *clkTrans (uTime: TM_ClockTime): TM_ClockTime* shall accept a 24-hour local or UTC time and return the equivalent time of day expressed in terms of the specified clock.

b) *utcTrans (clkTime: TM_ClockTime): TM_ClockTime* shall accept a time of day expressed in terms of the specified clock and return the equivalent time of day in 24-hour local or UTC time.

5.3.3 Temporal coordinate systems

Specifying temporal position in terms of calendar date and time of day complicates the computation of distances between points and the functional description of temporal operations. A temporal coordinate system may be used to support applications of this kind. A temporal coordinate system shall be based on a continuous interval scale defined in terms of a single time interval.

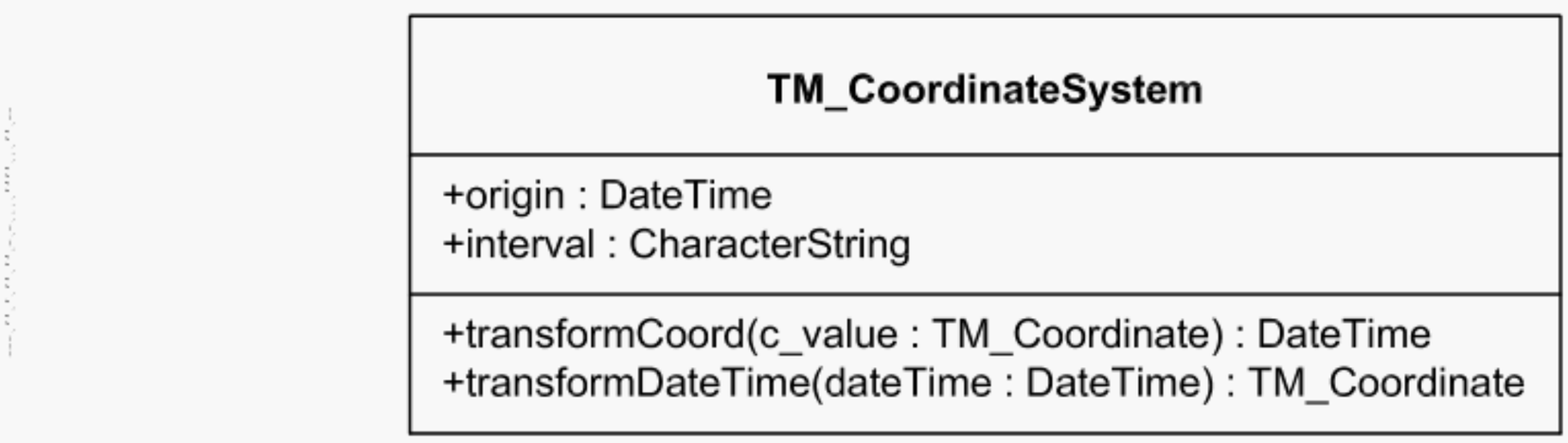

Figure 9 — Temporal coordinate system

TM_CoordinateSystem (see Figure 9) shall contain two attributes:

a) *origin: DateTime* shall provide the origin of the scale. The origin shall be specified in the Gregorian calendar with time of day in UTC. The DateTime may be truncated to the appropriate level of resolution.

b) *interval: CharacterString* shall return the name of a single unit of measure used as the base interval for the scale. The time interval may be selected as appropriate for the application, but it shall be one of those units of measure for time specified by ISO 31-1, or a multiple of one of those units, as specified by ISO 1000.

TM_CoordinateSystem shall support two operations:

a) *transformCoord (c_value: TM_Coordinate): DateTime* shall accept a value of a coordinate within this temporal coordinate system and return the equivalent DateTime in the Gregorian Calendar and UTC.

b) *transformDateTime (dateTime: DateTime): TM_Coordinate* accepts a DateTime in the Gregorian Calendar and UTC, and returns the equivalent TM_Coordinate.

5.3.4 Ordinal temporal reference systems

In a number of applications of geographic information — geology and archaeology, for example — relative position in time is known more precisely than duration. The order of events in time can be well established, but the magnitude of the intervals between them can not be accurately determined. An ordinal temporal reference system is appropriate in such cases.

An ordinal temporal reference system is based on an ordinal scale. In its simplest form, an ordinal temporal reference system is an ordered series of events. Generally, a specific series of events is associated with a single location. Temporal relationships between different locations can be determined only to the degree that events at one location can be correlated with events at other locations on the basis of non-temporal characteristics of the events. Such correlation can be used to develop a more broadly based temporal reference system defined in terms of periods within which similar events have occurred. The term ordinal era is used in this standard to refer to such a period.

An ordinal temporal reference system consists of a set of ordinal eras (see Figure 10). Ordinal reference systems are often hierarchically structured such that an ordinal era at a given level of the hierarchy includes a sequence of coterminous shorter ordinal eras.

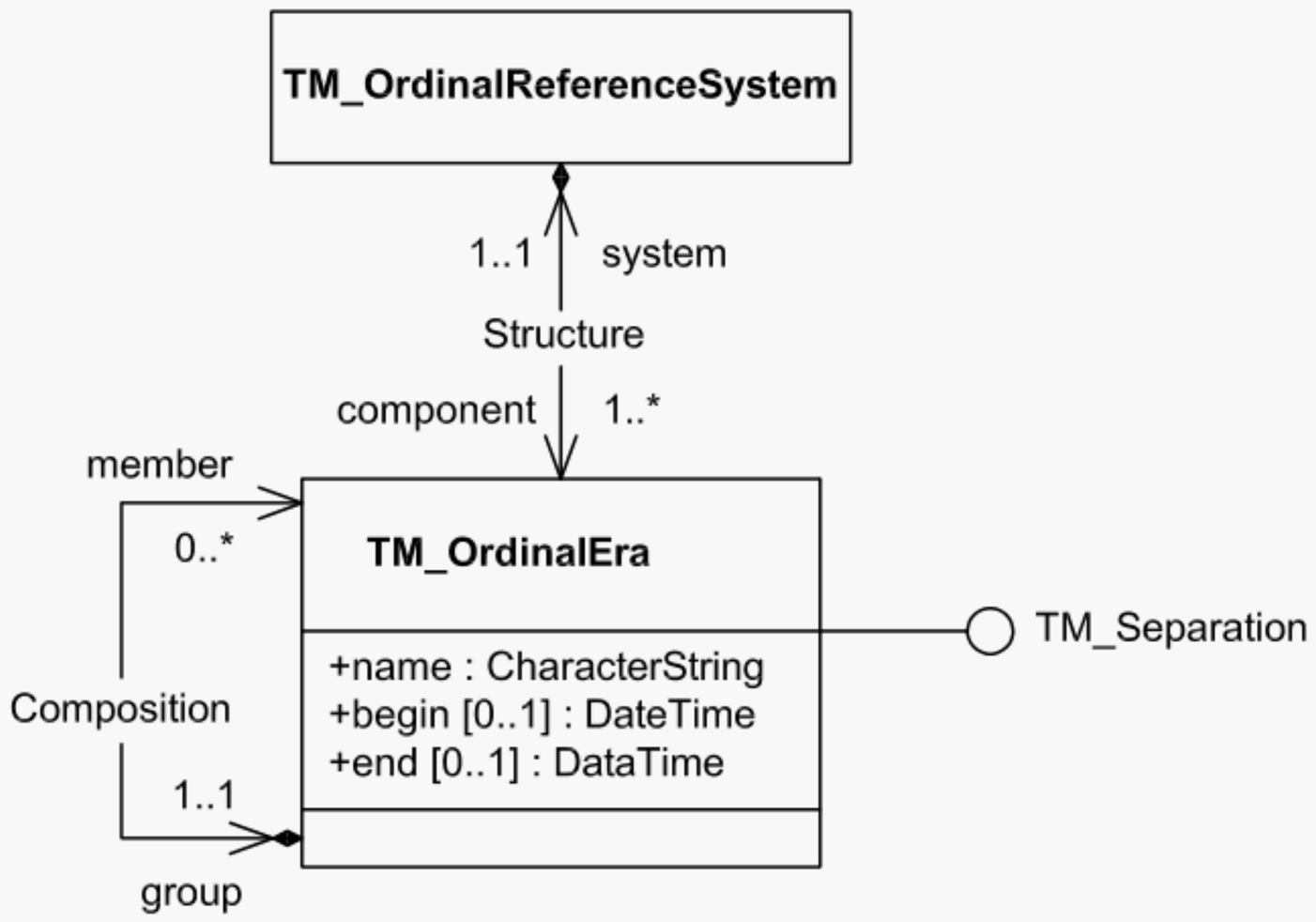

Figure 10 — Ordinal temporal reference system

TM_OrdinalReferenceSystem provides only the attributes inherited from TM_ReferenceSystem. The association Structure points to the sequence of TM_OrdinalEras that make up the highest level of the hierarchy.

TM_OrdinalEra contains three attributes:

a) *name: CharacterString* shall be a string that uniquely identifies the ordinal era within the TM_OrdinalReferenceSystem.

b) *begin: DateTime* may provide the temporal position at which the ordinal era began, if it is known. The TM_TemporalPosition shall be specified as a DateTime in the Gregorian calendar with time of day in UTC. The DateTime may be truncated to the appropriate level of resolution, as specified in ISO 8601.

c) *end: DateTime* may provide the temporal position at which the ordinal era ended, if it is known. The TM_TemporalPosition shall be specified as a DateTime in the Gregorian calendar with time of day in UTC. The DateTime may be truncated to the appropriate level of resolution, as specified in ISO 8601.

TM_OrdinalEra may support the interface TM_Separation (see 5.2.3.6).

A TM_OrdinalEra may be composed of a sequence of shorter TM_OrdinalEras identified by the association Composition.

5.4 Temporal position

5.4.1 Introduction

The method for identifying a temporal position is specific to each type of temporal reference system. The preferred reference system for use with geographic information is the combination of the Gregorian calendar with Coordinated Universal Time (5.3.1). ISO/TS 19103 defines data types for expressing dates as character strings that comply with ISO 8601. ISO 8601 specifies the use of the Gregorian calendar and UTC. This International Standard defines data types that shall be used to specify temporal positions in other temporal reference systems.

5.4.2 TM_Position

TM_Position is a union class that consists of one of the data types listed as its attributes. Date, Time, and DateTime are basic data types defined in ISO/TS 19103. They comply with ISO 8601 encoding of dates and times as character strings. These data types may be used for describing temporal positions referenced to the Gregorian calendar and UTC. TM_TemporalPosition and its subtypes shall be used for describing temporal positions referenced to other temporal reference systems. The data types defined in 5.4.4 specify numeric values for dates and times. They may be used for temporal positions referenced to any calendar or clock, including the Gregorian calendar and UTC.

5.4.3 TM_TemporalPosition

TM_TemporalPosition has four subclasses (see Figure 11), an association with TM_ReferenceSystem, and one attribute.

Attribute:

a) *indeterminatePosition: TM_IndeterminateValue* is an optional attribute. This attribute provides the only value for TM_TemporalPosition unless a subtype of TM_TemporalPosition is used as the data type. When this attribute is used with a subtype of TM_TemporalPosition, it provides a qualifier to the specific value for temporal position provided by the subtype.

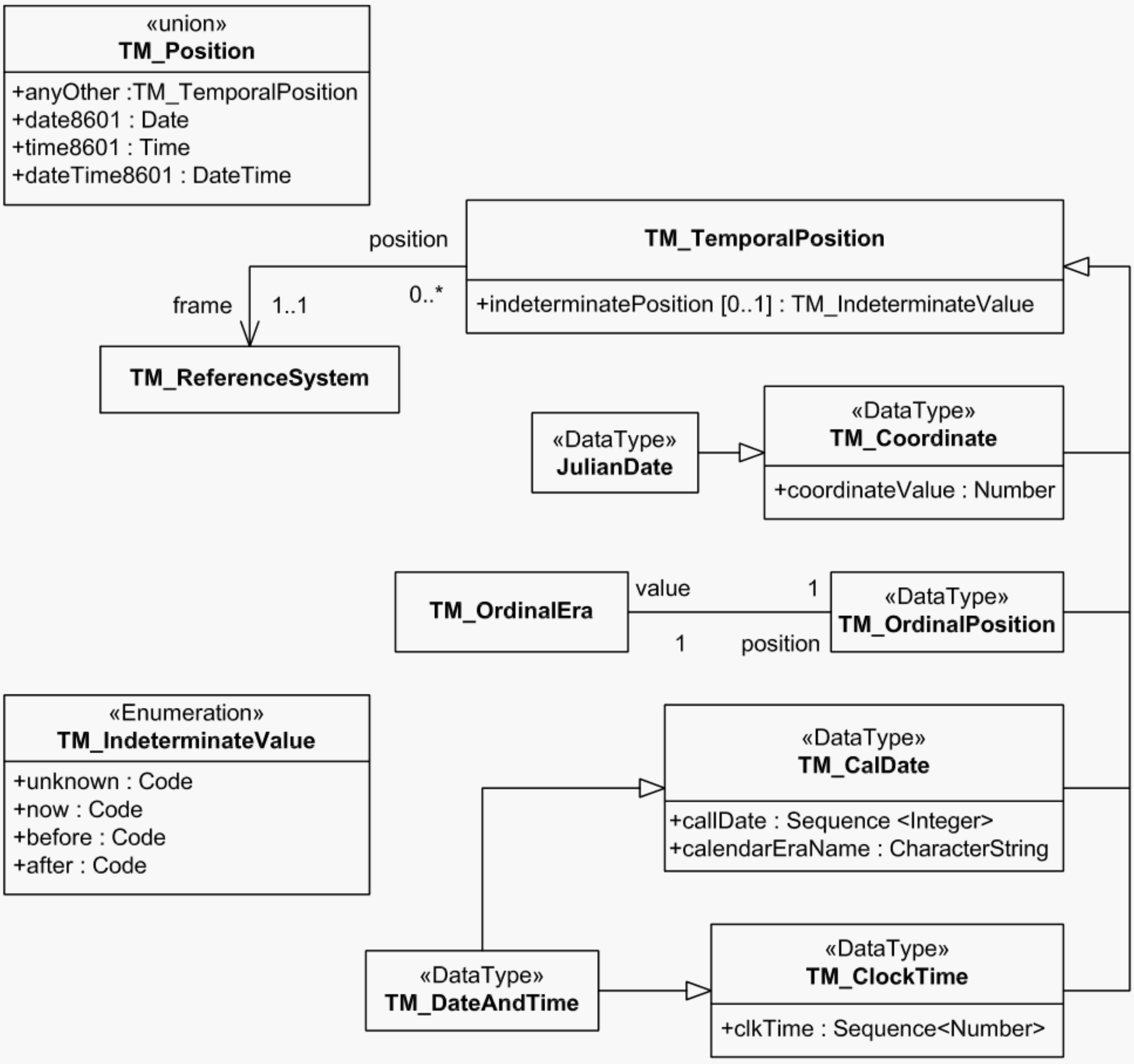

Figure 11 — Data types for temporal position

The enumerated data type TM_IndeterminateValue provides 4 values for indeterminate positions.

a) “unknown” shall be used with the parent class TM_TemporalPosition to indicate that no specific value for temporal position is provided.

b) “now” shall be used with any subtype of TM_TemporalPosition to indicate that the specified value shall be replaced with the current temporal position whenever the value is accessed.

c) “before” shall be used with any subtype of TM_TemporalPosition to indicate that the actual temporal position is unknown, but it is known to be before the specified value.

d) “after” shall be used with any subtype of TM_TemporalPosition to indicate that the actual temporal position is unknown, but it is known to be after the specified value.

Association:

a) The association *Reference* connects the TM_TemporalPosition to a TM_ReferenceSystem. Every TM_TemporalPosition shall be associated with a TM_ReferenceSystem. This association need not be explicit at the instance level. If not specified, it is assumed to be an association to the Gregorian calendar and UTC (see 5.3.1); it can also be identified in the attribute type definition in a feature catalogue, or in the metadata for a data set.

5.4.4 Position referenced to calendar and clock

5.4.4.1 Calendar date

TM_CalDate is a data type that shall be used to identify temporal position within a calendar. TM_CalDate has two attributes:

a) *calendarEraName: CharacterString* provides the name of the calendar era to which the date is referenced.

b) calDate: Sequence <Integer> provides a sequence of positive integers in which the first integer identifies a specific instance of the unit used at the highest level of the calendar hierarchy, the second integer identifies a specific instance of the unit used at the next lower level in the hierarchy, and so on. The format defined in ISO 8601 for dates in the Gregorian calendar may be used for any date that is composed of values for year, month and day.

EXAMPLE In the Gregorian calendar, the sequence 1999, 09, 03 identifies a temporal position as being the third day of the ninth month of the year 1999. This would be expressed in ISO 8601 format as 19990903.

5.4.4.2 Clock time

TM_ClockTime is a data type that shall be used to identify a temporal position within a day. Because TM_TemporalPosition cannot by itself completely identify a single temporal position, it shall be used with TM_CalDate for that purpose. It may be also used to identify the time of occurrence of an event that recurs every day. TM_ClockTime has a single attribute:

a) *clkTime: Sequence <Number>* provides a sequence of positive numbers with a structure similar to that of *calDate*. The first number integer identifies a specific instance of the unit used at the highest level of the clock hierarchy, the second number identifies a specific instance of the unit used at the next lower level, and so on. All but the last number in the sequence shall be integers; the last number may be integer or real.

EXAMPLE In modern 24-hour time the sequence 22, 15, 30.5 identifies a temporal position 30,5 s after the start of the 15th minute of the 22nd hour. This would be expressed in ISO 8601 format as 221530.5.

5.4.4.3 Calendar date with clock time

TM_DateAndTime is subtype of both TM_CalDate and TM_ClockTime; it inherits the attributes of both to provide a single data type for identifying a temporal position with a resolution of less than a day.

5.4.5 Position referenced to a temporal coordinate system

5.4.5.1 TM_Coordinate

TM_Coordinate is a data type that shall be used for identifying temporal position within a temporal coordinate system. TM_Coordinate has a single attribute:

a) *CoordinateValue: Number* holds the distance from the scale origin expressed as a multiple of the standard interval associated with the temporal coordinate system.

5.4.5.2 Julian date

The Julian day numbering system is a temporal coordinate system that has its origin at noon on 1 January 4713 BC in the Julian proleptic calendar. The Julian day number is an integer value; the Julian date is a decimal value that allows greater resolution.

5.4.6 Position referenced to an ordinal temporal reference system

Within an ordinal temporal reference system, the temporal position of an instant is within the shortest ordinal era (at the lowest level) in which it occurs. TM_OrdinalPosition is a data type that shall be used for identifying temporal position within an ordinal temporal reference system. TM_OrdinalPosition has one attribute:

a) *ordinalPosition: Reference <TM_OrdinalEra>* provides a reference to the ordinal era in which the instant occurs.

EXAMPLE Table 1 shows part of the geologic time scale, which is an ordinal temporal reference system. The temporal position of an instant in the Cenozoic Era should be identified by the name of the Epoch within which it falls, while the temporal position of an instant in the Mesozoic Era would be identified by the Period within which it falls.

Table 1 — Part of the geologic time scale

Eras	Periods	Epochs
Cenozoic	Quaternary	Holocene
		Pleistocene
	Tertiary	Pliocene
		Miocene
		Oligocene
		Eocene
		Paleocene
Mesozoic	Cretaceous	
	Jurassic	
	Triassic	

5.5 Time and components of geographic information

5.5.1 Temporal aspects of geographic information components

The general feature model described in ISO 19109 provides a metamodel for geographic information. It identifies metaclasses for feature attributes, feature operations, and feature associations as shown in Figure 12. Each of these components may have a temporal aspect. ISO 19109 provides rules for instantiating these metaclasses in an application schema. Subclauses 5.5.2 through 5.5.4 identify temporal subtypes of some of the metaclasses in the General Feature Model. They also provide extensions to the rules for use when these components involve temporal information. Annex B provides examples of the way that these classes and rules should be used in an application schema.

ISO 19110 specifies rules for compiling feature catalogues, which include definitions of feature types, feature attributes, feature operations, and feature associations. Subclauses 5.5.2 through 5.5.4 specify requirements for defining these components when they have a temporal aspect.

ISO 19115 specifies a standard set of metadata elements for use with geographic information, and provides a mechanism for defining additional metadata elements appropriate to specific applications or profiles. Subclause 5.5.5 identifies requirements for defining metadata elements that have a temporal aspect.

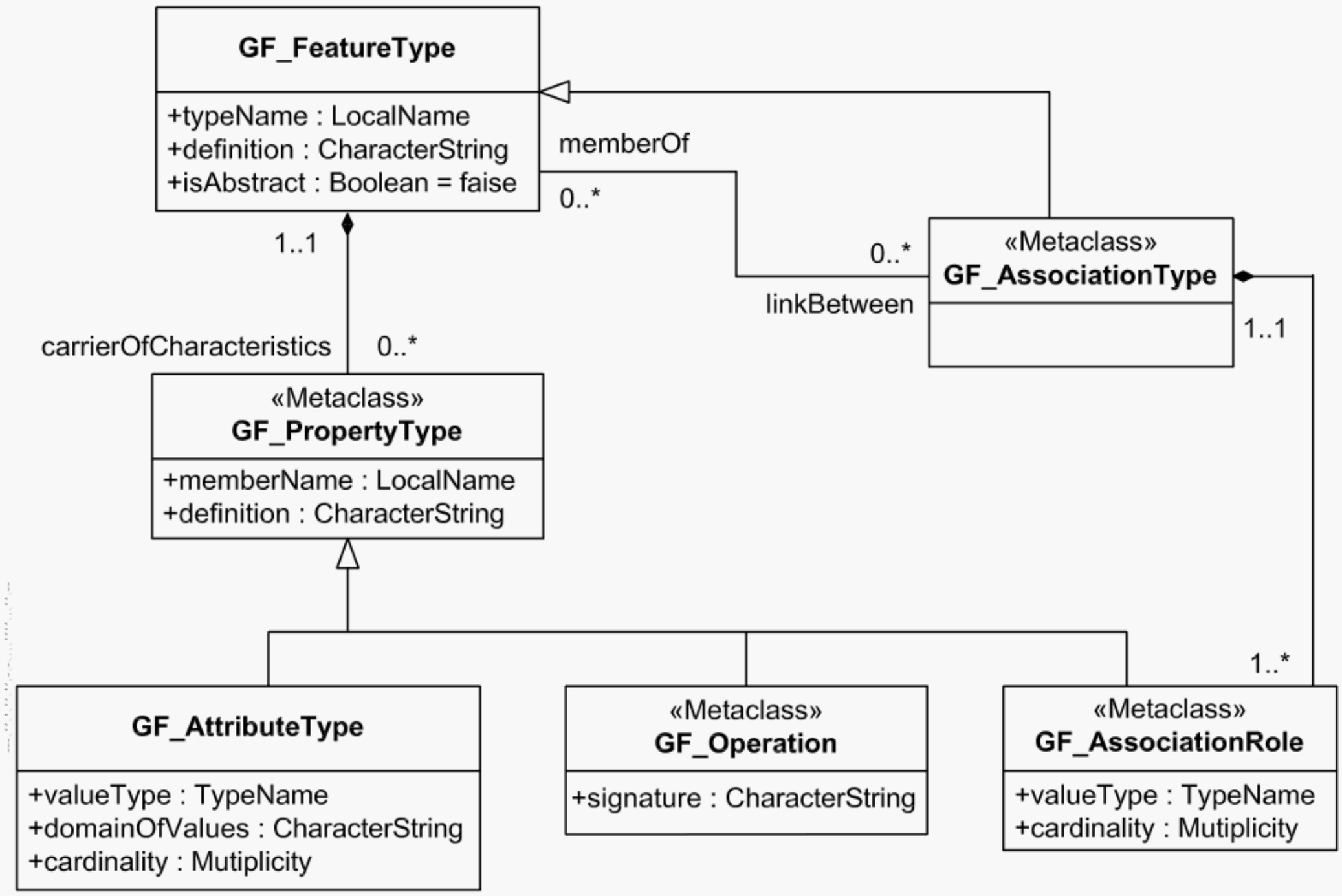

Figure 12 — Metaclasses of the general feature model

5.5.2 Temporal feature attributes

A temporal feature attribute describes a feature characteristic that is associated with a temporal position. In accordance with the rules for application schemas defined in ISO 19109, a GF_TemporalAttributeType shall normally be instantiated in an application schema as an attribute of a class that is itself an instantiation of the GF_FeatureType metaclass. Under the conditions specified in ISO 19109, it may be instantiated as a UML class that is associated with the feature type class for which it is an attribute.

Static temporal characteristics are of two kinds: events and states (see Figure 13).

a) An event is an action that occurs at an instant. In fact, almost every event occupies a short interval of time, but when that interval is short relative to the resolution of the scale of measurement, it is specified as an instant. The GF_AttributeType.valueType of an event shall be either a TM_Instant, a TM_Node, or a TM_TemporalPosition.

b) A state is a condition — a characteristic of a feature or data set that persists for a period. That characteristic may be represented by a feature attribute or metadata element. A GF_TemporalAttributeType that describes a state may be instantiated in two ways. In the simple case, the GF_AttributeType shall be instantiated as an attribute of a class that represents a feature type. Its GF_AttributeType.valueType shall be either a TM_Period or a TM_Edge. When more information is needed, a GF_TemporalAttributeType that represents a state shall be instantiated as a UML class. That class shall be a subtype of TM_Period that inherits the associations Beginning and Ending, or a subtype of TM_Edge that inherits the associations Initiation and Termination. The characteristics of the state shall be described by one or more attributes of the class. Its recurrence shall be indicated by the multiplicity at the attribute end of its association with the feature type class. Often, a change in state is associated with an event that initiates or terminates the state. That event shall be identified by an attribute of the class that represents the state.

An event may recur at multiple instants; a state may also recur at multiple times. The FeatureAttributeType.cardinality shall specify the number of recurrences that the application schema allows.

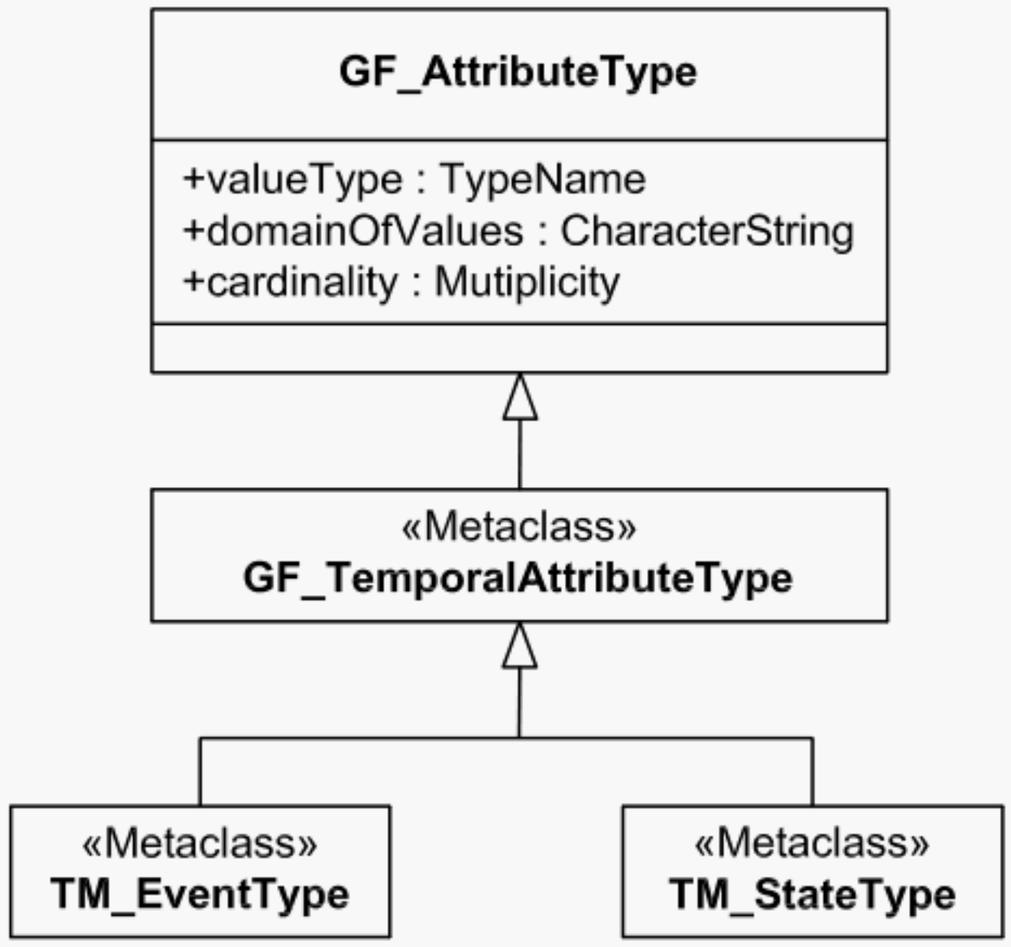

Figure 13 — Temporal feature attribute types

Events or states often recur on a regular basis. The duration of the interval between two successive occurrences of an event or state is its periodic time. When a temporal feature attribute describes a recurrent phenomenon, it shall be instantiated in an application schema as a UML class that is associated with a feature type class. This class shall have at least two attributes: one that identifies a specific instant at which the event occurs or a period during which the thematic value of the attribute applies, and one which identifies the periodic time between occurrences of the event or state. The data type TM_Duration (see 5.2.3.7) shall be used to express the value for periodic time.

ISO 19110 specifies elements that shall be included in the description of a feature attribute within a feature catalogue. When a GF_TemporalAttributeType is an event, the Feature Attribute Definition of the attribute shall identify the action and the resolution to which its temporal position is specified. When a GF_TemporalAttributeType is a state, the Feature Attribute Definition of the attribute shall identify the characteristics of the state. In the case of a temporal feature attribute, the value for the Feature Attribute Value Domain Type is “not enumerated”. The Feature Attribute Value Domain, the Feature Attribute Value Data Type, and the Feature Attribute Value Measurement Unit are determined by the temporal reference system that is the basis for describing the temporal position of the instant or period. If the temporal reference system is not a combination of the Gregorian calendar and UTC, the description of the Feature Attribute Value Domain shall identify the temporal reference system that is used. Both the Feature Attribute Value Data Type and the Feature Attribute Value Measurement Unit shall be consistent with the temporal reference system.

5.5.3 Temporal feature operations

Temporal feature operations are dynamic — they describe the way in which the values of one or more aspects of a feature change as a function of time. A temporal feature operation shall be instantiated in an application schema as an operation of a feature type class for which time is included as an input parameter in the GF_Operation.signature.

EXAMPLE Given a feature type class RoadIntersection, traffic flow through the intersection could be described by a UML operation such as:

TrafficFlow (tMin: Integer, tMax: Integer, time:TM_ClockTime): Integer

ISO 19110 specifies requirements for defining feature operations. In the case of a temporal feature operation, the Feature Operation Textual Description shall describe the way in which the return value of the operation changes as a function of time, and how it is influenced by temporal characteristics. It shall also identify the temporal reference system within which time is measured. The Feature Operation Formal Definition shall include time as one of the independent variables.

5.5.4 Time and feature associations

5.5.4.1 Temporal aspects of feature associations

Feature associations may involve time in two ways. Some feature associations exist because of the temporal characteristics of the related feature instances. Other feature associations, which may exist for variety of reasons, have their own temporal characteristics as associations.

EXAMPLE The association between two features that exist at the same time illustrates the first case. Such a temporal association has no other characteristics, and is not dependent on the feature types involved. An example of the second case is the association between two tracts of land that are held by the same owner for some period. In this case, the dominant element of the association is the common ownership; the temporal element is a subordinate characteristic of the association.

Temporal feature associations are those of the first type — those that exist because of the temporal characteristics of the related feature instances.

5.5.4.2 Temporal feature associations

5.5.4.2.1 Introduction

A temporal feature association is an explicit description of an association between the life spans of the features linked by the association. It is possible to derive a temporal feature association by using the operation TM_Order.relativePosition to compare the life spans of two features, assuming that each feature has an attribute that identifies its life span. If temporal feature associations are important to an application, the application schema should assign a life span attribute to each feature type that it specifies. The attribute lifeSpan:TM_Period may be used when temporal positions are described in terms of either a calendar and clock or a temporal coordinate reference system. When temporal positions are described in terms of an ordinal temporal reference system, the attribute lifeSpan:TM_Edge should be used, and the application schema should instantiate TM_TopologicalComplex as a non-linear graph that includes all of the TM_Edges that represent life spans. Explicit temporal feature associations may be specified for applications that do not support the TM_Order interface.

This International Standard specifies two subtypes of temporal feature associations: simple temporal feature associations and succession (see Figure 14).

5.5.4.2.2 Simple temporal feature associations

A simple temporal feature association identifies the relative position in time of two or more features, and nothing else. In principle, this type of association exists between all feature instances, but cannot be predicated for any particular feature type.

According to the rules for application schemas defined in ISO 19109, feature associations shall normally be instantiated as associations between UML classes that represent feature types. However, purely temporal feature associations are usually independent of feature type. Instantiation of temporal feature associations as associations between feature type classes may be appropriate if an application is only concerned with temporal feature associations between certain types of features. This would be inefficient for an application schema that needs to carry information about some type of temporal association between instances of all feature types. It would be better for an application schema to support derivation of these associations as described in 5.5.4.2.1. An alternative would be to instantiate a feature class that is a supertype for all feature type classes in the schema, and instantiate the feature association as a self-referent association at that level.

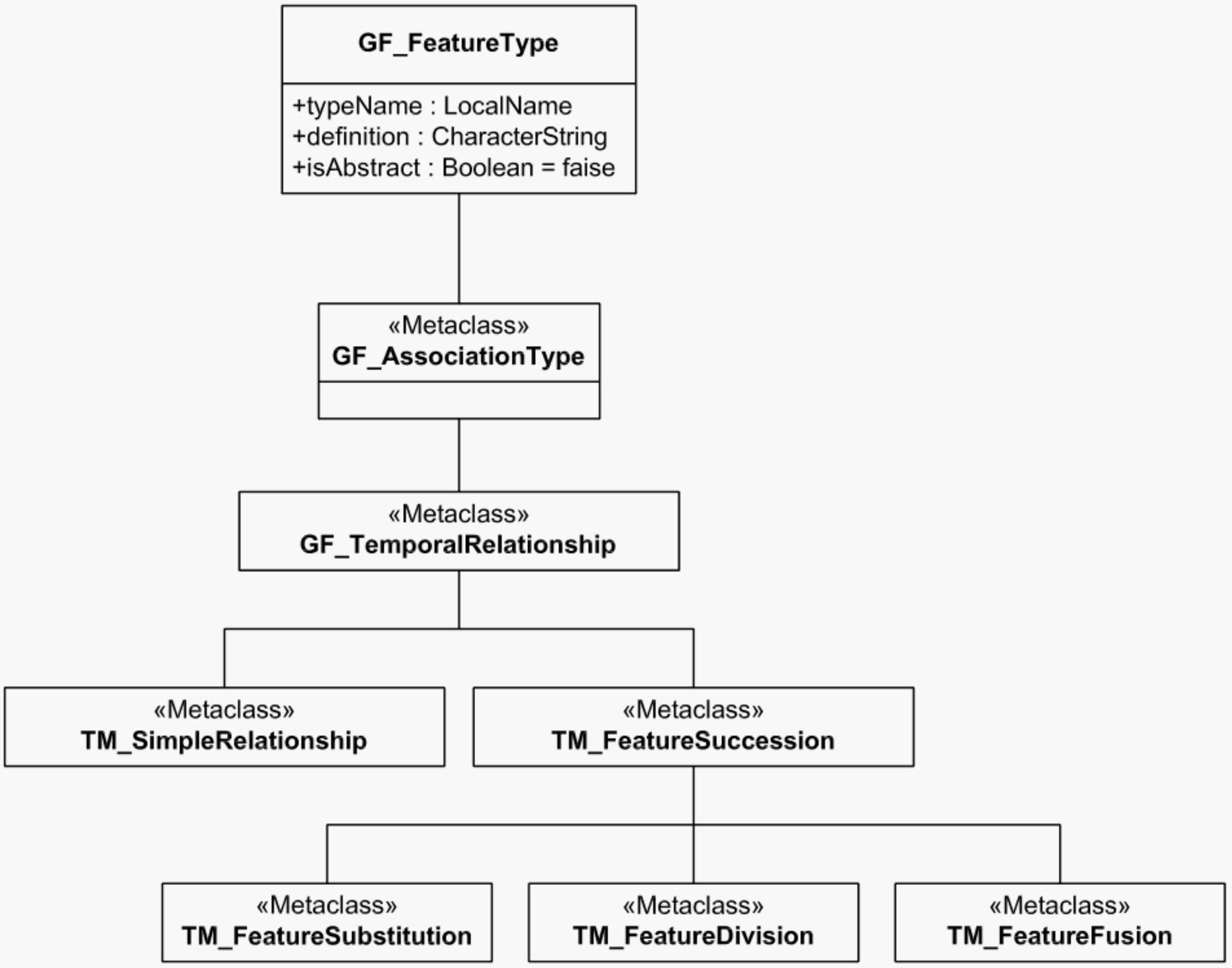

Figure 14 — Classification of temporal feature associations

5.5.4.2.3 Feature succession

Feature succession is the replacement of one set of feature instances by another set of feature instances. A set may include one or more feature instances. Replacement implies that the life spans of the feature instances in the first set come to an end at the instant when the life spans feature instances in the second set begin. There are both spatial and temporal aspects to feature succession, in that the feature instances in the association occupy the same spatial location, at different times and in a particular order.

Feature succession is not always type dependent. That is, the type of a feature instance is not always a predictor of the type of the feature instance that replaces it. Feature succession can be modelled at the generic feature level, but not always at the feature type level.

There are three kinds of feature succession: feature substitution, feature division, and feature fusion. Feature substitution is the replacement of one feature instance by another feature instance of the same or a different feature type. It establishes a one-to-one association between two feature instances. Feature division occurs when a single feature instance separates into two or more feature instances of the same type. It establishes a one-to-many association between feature instances. Feature fusion occurs when two or more instances of the same feature type merge into a single feature instance. It establishes a many-to-one association between feature instances. A single event may result in a form of succession that is a combination of these types. Four of these combinations are possible:

— division and substitution;

— fusion and division;

— fusion, substitution and division;

— substitution and fusion.

EXAMPLE Clearing part of a woodland and replacing it with a parking lot is an example of “division and substitution”. Clearing a woodland adjacent to a pasture and combining it with the pasture is an example of “substitution and fusion”.

NOTE Change in the characteristics of a single feature is not, in itself, characterized as feature succession. For example, consider a feature type Building that has an attribute numberOfOccupants. The value of this attribute might be updated on a regular basis, but this would not be considered a replacement of one instance of Building by another instance of Building. The degree of change that is necessary before one instance of a feature type is considered to have replaced an earlier instance of the same feature type depends upon the application. As a general rule, replacement may be considered to have occurred when the feature identifier changes.

Succession associations may be derived by using the operation TM_Order.relativePosition together with one or more of the spatial operations defined in ISO 19107.

Temporal feature associations of the feature succession type may also be instantiated in an application schema as UML associations between feature type classes, or as self-referent associations of a generic feature class. The names, roles, and multiplicities shall be different for each type of succession. The role names shall indicate the order in which one feature succeeded another. To include the time at which succession occurred, an application schema shall represent the succession association as a UML association class with an attribute that identifies the time of occurrence.

5.5.4.3 Temporal characteristics of feature associations

Temporal characteristics of feature associations usually provide information about the instant at which an association began or ended, or about the period for which it persisted. Like temporal feature attributes or temporal metadata elements, they describe events or states. Feature associations that have temporal characteristics shall be instantiated as UML association classes. The temporal characteristic shall be represented as an attribute that instantiates the TM_Event or TM_State metaclass (5.5.2).

5.5.5 Temporal metadata elements

ISO 19115 defines a set of standard metadata elements for geographic information. It also specifies a methodology for defining additional metadata elements within an application schema (see Figure 15). Temporal metadata elements are similar to temporal feature attributes. Both describe a static characteristic — an event or a state — associated with a temporal position. They differ in that feature attributes describe characteristics of a real world object as abstracted in the feature instance, while metadata elements describe characteristics of data. The scope of the data described by a metadata element may range from a collection of data sets to a single characteristic of a feature.

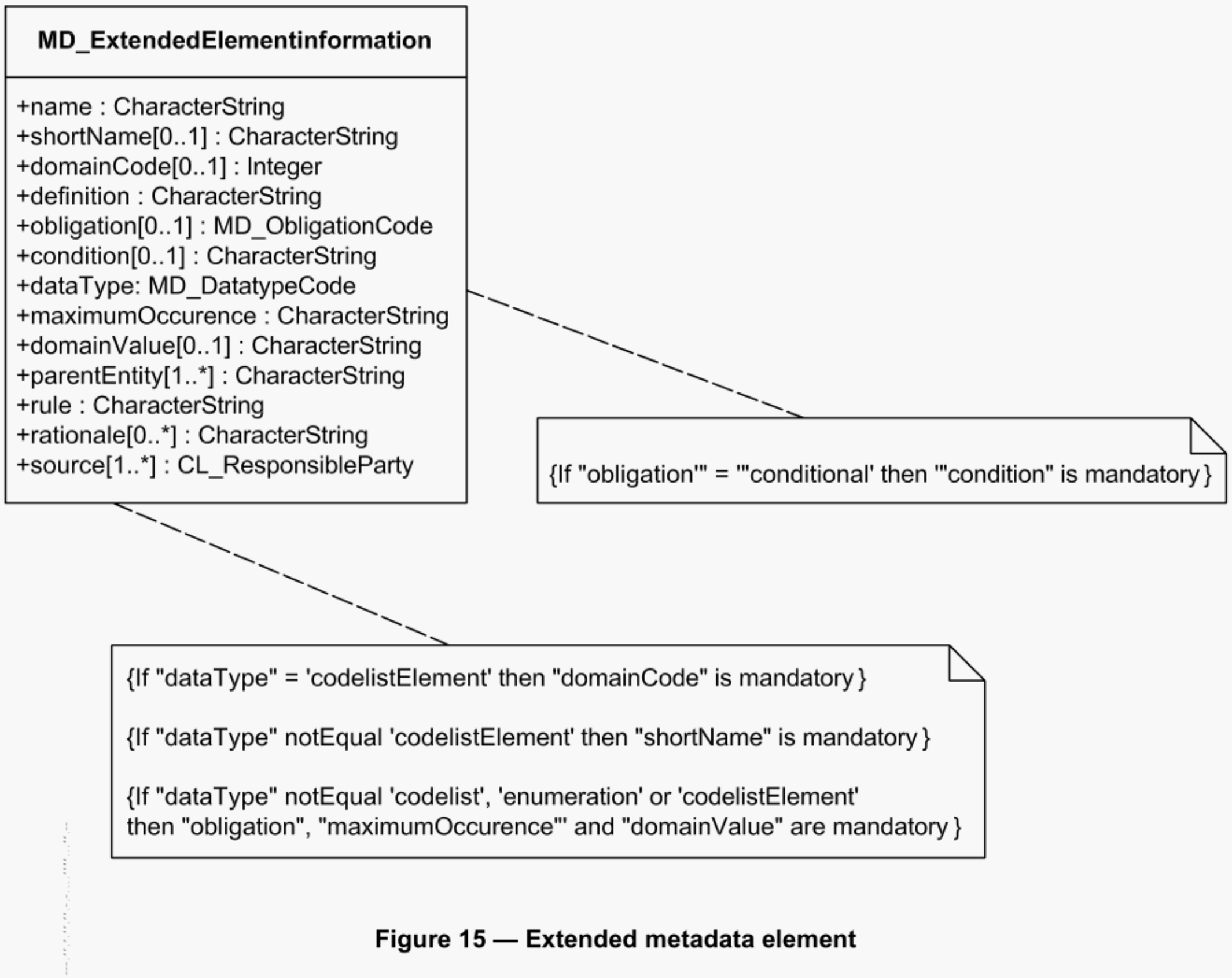

Figure 15 — Extended metadata element

When a temporal metadata element describes an event, the name and the definition shall identify the action and the resolution to which its temporal position is specified. The value of the metadata element shall be an instance of TM_Instant, of TM_TemporalPosition, or of one of the subtypes of TM_TemporalPosition.

When a temporal metadata element describes a state, the name and the definition shall describe the characteristics of the state. The value of the metadata element shall be a TM_Period.

If the metadata element is not referenced to a combination of the Gregorian calendar and UTC, the definition shall identify the temporal reference system that is used.

Annex A
(normative)

Abstract test suite

A.1 Application schemas for data transfer

a) Test Purpose: Verify that an application schema for data transfer defines temporal attributes and temporal associations of features and temporal metadata elements in compliance with specified requirements.

b) Test Method: Inspect the presentation of the temporal attributes and temporal associations of features included in the application schema to ensure that the definitions satisfy requirements for the use of temporal objects to represent temporal attributes or their values. Ensure that any temporal metadata elements defined in the application schema satisfy requirements. Ensure that all required attributes and associations of temporal objects are implemented, and that optional attributes or associations are implemented in compliance with requirements. Verify that required data types are used for values of temporal position.

c) Reference: 5.2, 5.4, 5.5.2, 5.5.4.2 and 5.5.5.

d) Test Type: Basic.

A.2 Application schemas for data with operations

a) Test Purpose: Verify that an application schema that supports operations on data satisfies the requirements of A.1, that it defines temporal feature operations, and that it implements operations of temporal objects in compliance with specified requirements.

b) Test Method: Inspect the presentation of temporal feature operations included in the application schema to ensure that they satisfy requirements. Inspect the presentation of the temporal attributes of features included in the application schema to ensure that any operations of temporal objects used to represent temporal attributes or their values are implemented in compliance with requirements.

c) Reference: A.1, 5.2, 5.5.3 and 5.5.

d) Test Type: Basic.

A.3 Feature catalogues

a) Test Purpose: Verify that the feature catalogue defines temporal characteristics in compliance with specified requirements.

b) Test Method: Examine the catalogue. Check definitions for temporal feature attributes, temporal feature operations, or temporal feature associations comply with requirements. Ensure that the requirements for identifying and describing temporal reference systems are satisfied.

c) Reference: 5.3, 5.5.2, 5.5.3 and 5.5.4.2.

d) Test Type: Basic.

A.4 Metadata element specifications

a) Test Purpose: Verify that temporal metadata elements are defined in compliance with specified requirements.

b) Test Method: Inspect the metadata element definitions in the metadata specification to ensure that the definitions satisfy the requirements appropriate for events or states.

c) Reference: 5.5.5.

d) Test Type: Basic.

A.5 Metadata for data sets

a) Test Purpose: Verify that metadata for a data set provides required information about temporal reference systems.

b) Test Method: Inspect the metadata set. If the metadata does not indicate that all temporal positions included in the data that it describes are referenced to the Gregorian Calendar and UTC, check to see that it provides either a description or a citation to a description of each of the temporal reference systems used with the data. Evaluate each description to verify that it includes all required elements.

c) Reference: 5.3.

d) Test Type: Basic.

Annex B
(informative)

Use of time in application schemas

B.1 Temporal feature attributes

B.1.1 TM_GeometricPrimitive as a data type

Figure B.1 illustrates the use of a TM_GeometricPrimitive in an application schema. The feature type Building is represented by a UML class. It has one attribute called dateOfConstruction, which takes a value of data type TM_Instant, and a second attribute, called periodOfOccupancy, which takes a value of data type TM_Period. In this schema, the TM_GeometricPrimitives do not implement the operations defined at the interfaces TM_Order and TM_Separation. Since TM_Instant has only one attribute — TM_Instant.position: TM_TemporalPosition — which takes a value of data type TM_TemporalPosition, either that data type, or one of its subtypes, such as DateTime, could be used as the data type for Building.dateOfConstruction.

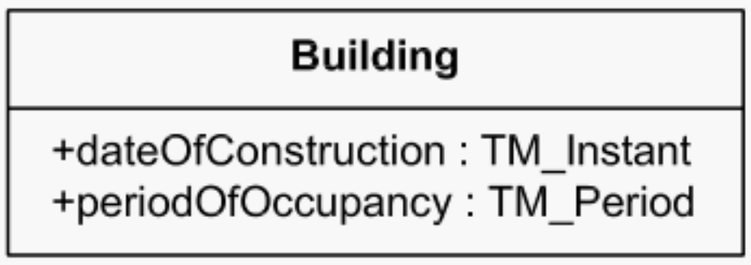

Figure B.1 — TM_GeometricPrimitives as data types

B.1.2 TM_GeometricPrimitive as a temporal attribute

Figure B.2 illustrates an alternative way of using a TM_GeometricPrimitive for a temporal feature attribute. In this case, the feature attribute periodOfOccupancy is represented as a UML class that is linked to Building by a UML association. PeriodOfOccupancy is a subtype of TM_Period. It inherits the association roles begin and end from TM_Period, but it restricts the data type in each case to DateTime. [At the TM_Period level, the data type for these attributes is TM_TemporalPosition.] It also inherits the interfaces TM_Order, from which it uses the operation relativePosition (other: TM_Primitive): TM_RelativePosition, and TM_Separation, from which it uses the operations Length(): TM_Duration and distance (other: TM_GeometricPrimitive): TM_Duration.

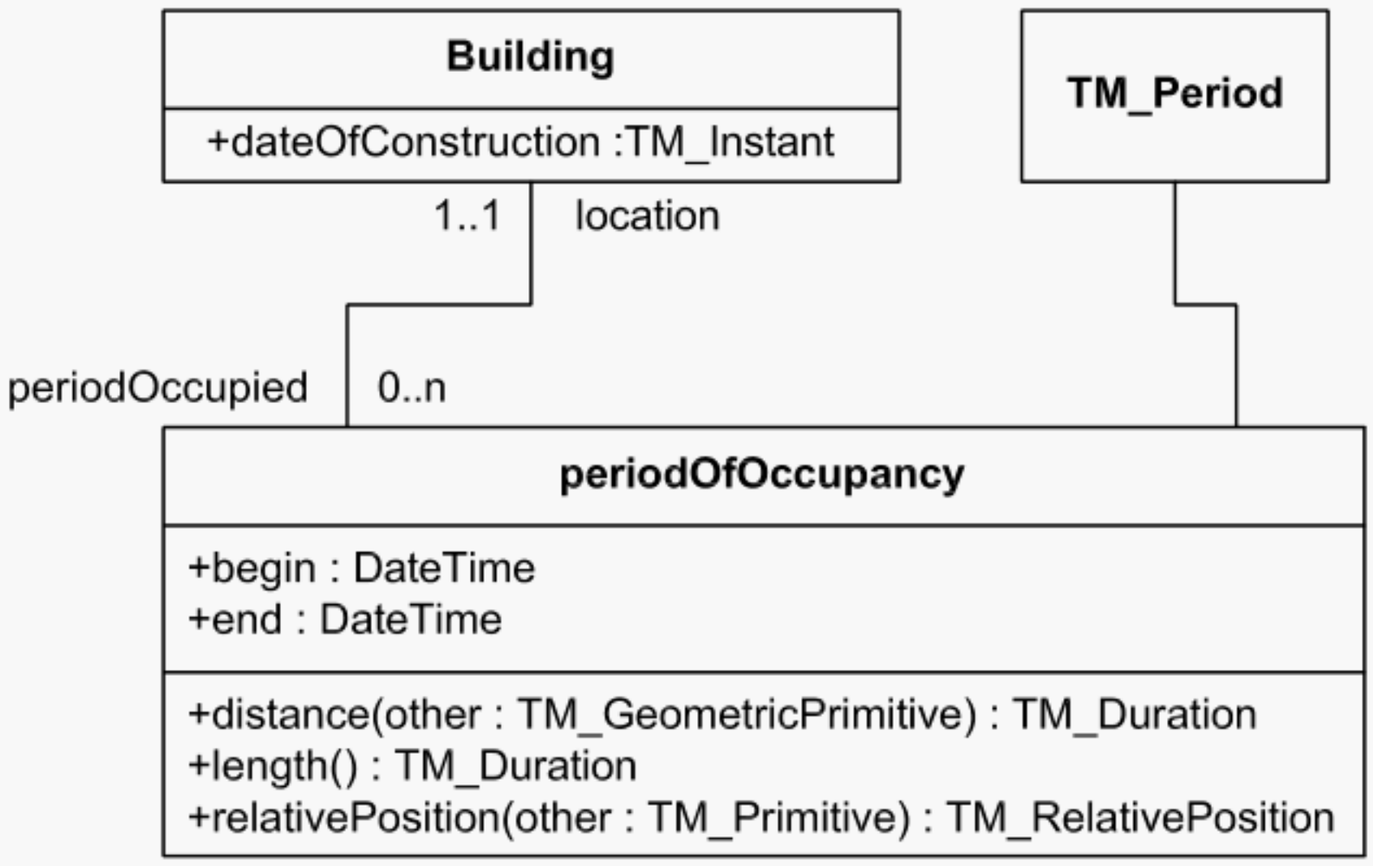

Figure B.2 — TM_GeometricPrimitives as temporal feature attributes

B.1.3 TM_TopologicalComplex as an attribute

Figure B.3 illustrates the use of a TM_TopologicalComplex as a temporal feature attribute. The history of an archaeological site is often described in terms of a series of periods during which the site was occupied by members of some cultural group, or unoccupied. The order of these periods is known, but the temporal position is not. This history can be described as a TM_ToplogicalComplex. The feature type ArchaeologicalSite, represented as a UML class, is associated with another UML class that represents the feature attribute SiteHistory. SiteHistory is a subtype of TM_TopologicalComplex. It is an aggregate of OccupancyIntervals, where each OccupancyInterval is a subtype of TM_Edge.

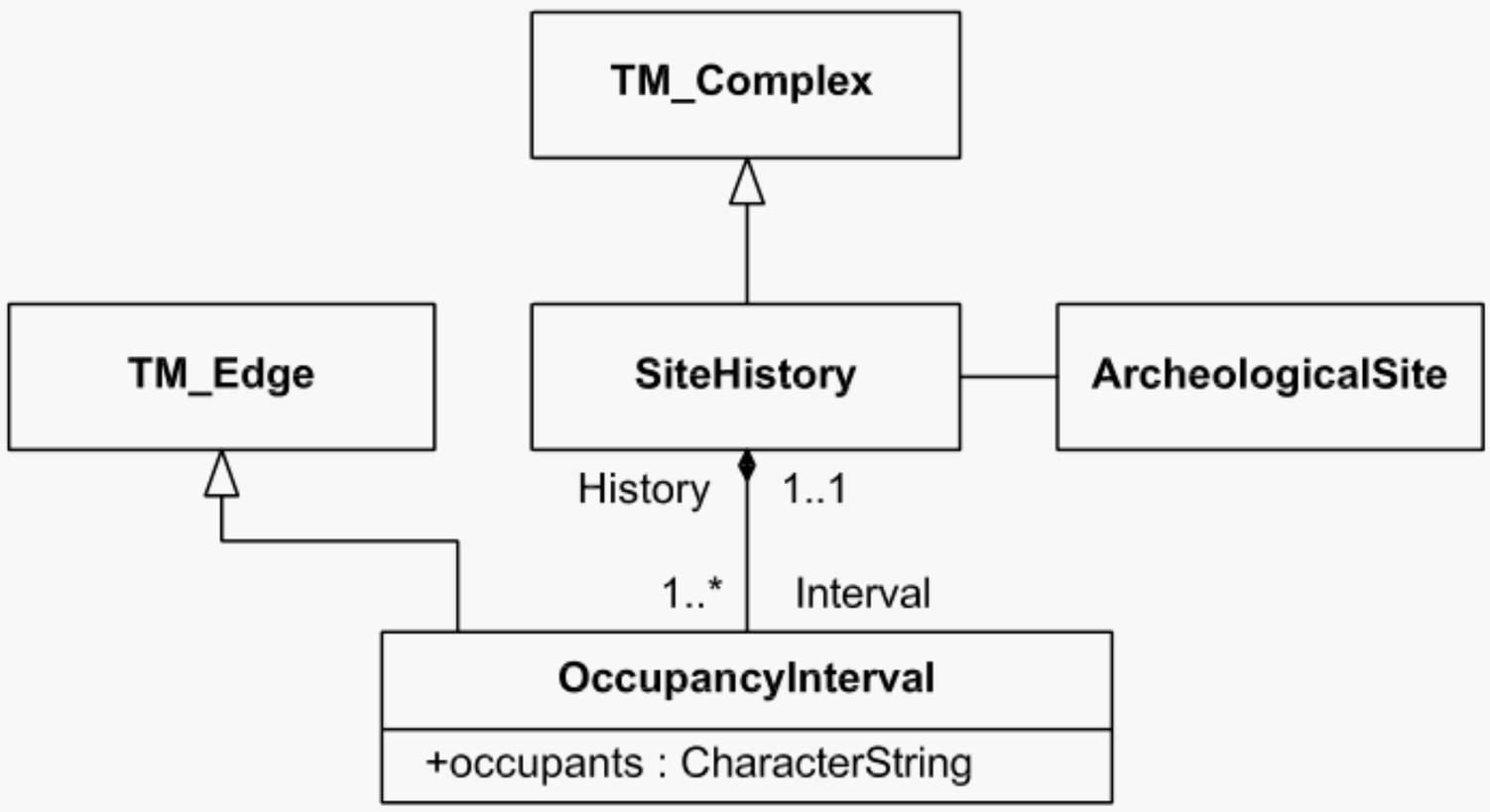

Figure B.3 — TM_TopologicalComplex as a temporal feature attribute

B.1.4 Recurring attribute values

Figure B.4 provides an example of a feature attribute that has a value that recurs cyclically. The feature type Field has an attribute CropRotation that is represented as a UML class with a multiplicity of 0 to n. There is one instance of this attribute type for each kind of crop that is planted in the field. CropRotation has three attributes:

— cropType identifies a kind of crop;

— firstPlanting identifies the date on which that crop was first planted in the field;

— rotationInterval identifies the periodic time between plantings of that crop.

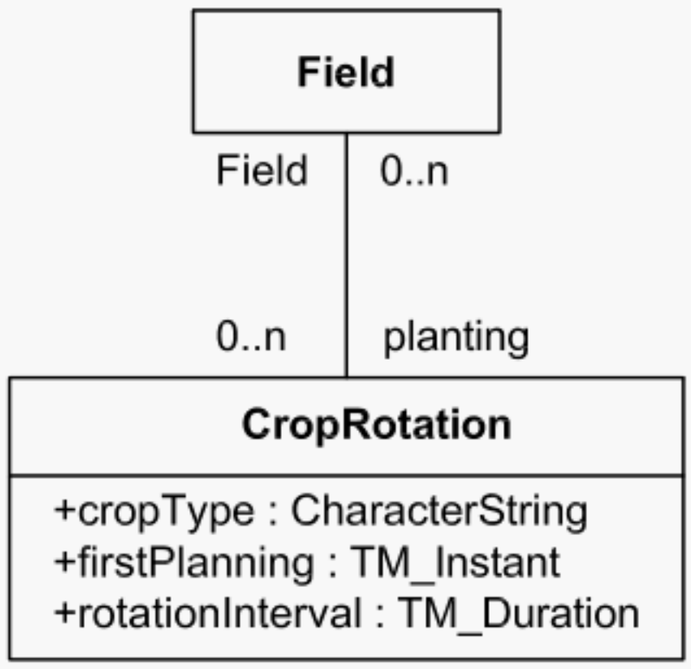

Figure B.4 — Cyclical attribute values

B.2 Temporal feature associations

B.2.1 Simple temporal associations

Figure B.5 is an example of a simple feature association implemented as a UML association. The schema indicates that roads exist before service stations exist.

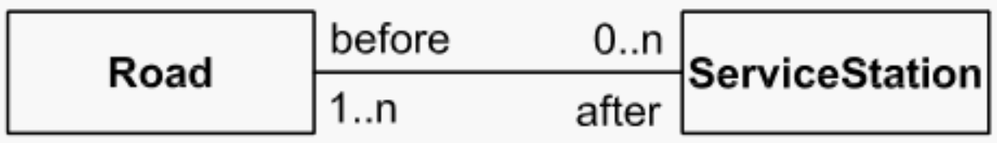

Figure B.5 — Explicit representation of a simple temporal association

The simple schema for buildings shown in Figure B.6 provides an example of the data required to derive simple feature associations by using the operation TM_Order.relativePosition.

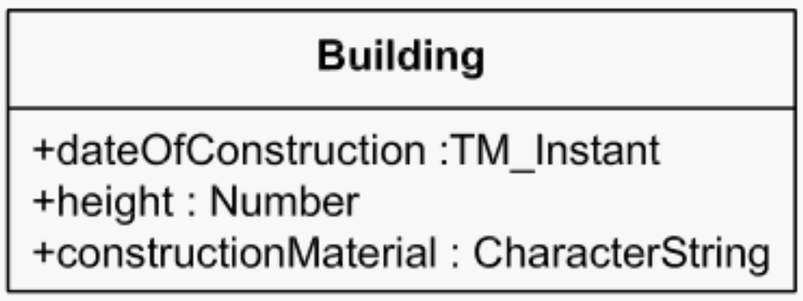

Figure B.6 — Building schema

Assume a data set built to this schema includes four instances of Building with the attribute values shown in Table B.1.

Table B.1 — Data set for Building schema

Building	constructionMaterial	height	dateOfConstruction: TM_Instant.position [as Date]
A	brick	45 m	1982
B	steel	4 m	1967
C	wood	8 m	1941
D	brick	30 m	1967

TM_Instant may use the operation relativePosition(other:TM_Primitive): TM_RelativePosition from the interface TM_Order. If that operation is performed for each instance of Building.dateOfConstruction: TM_Instant as source, with each of the other instances as targets (i.e. as values for the input parameter other), it will return the values listed in Table B.2.

Table B.2 — Simple temporal associations between instances of Building

Source Instance	Target Instance			
	A	B	C	D
A	Equal	After	After	After
B	Before	Equal	After	Equal
C	Before	Before	Equal	Before
D	Before	Equal	After	Equal

B.2.2 Feature succession

Figure B.7 is an example of a feature succession modelled as an association between feature type classes. It is an example of a type of ecological succession, known as old field succession, common in the eastern United States. The types occur on a single site in the sequence shown, if the site is left undisturbed. This illustrates the problem associated with modelling feature succession at the feature type level. At any time, fire, storm, or human intervention can interrupt this succession and move it back to an earlier stage, or replace the existing feature types with an unexpected feature type.

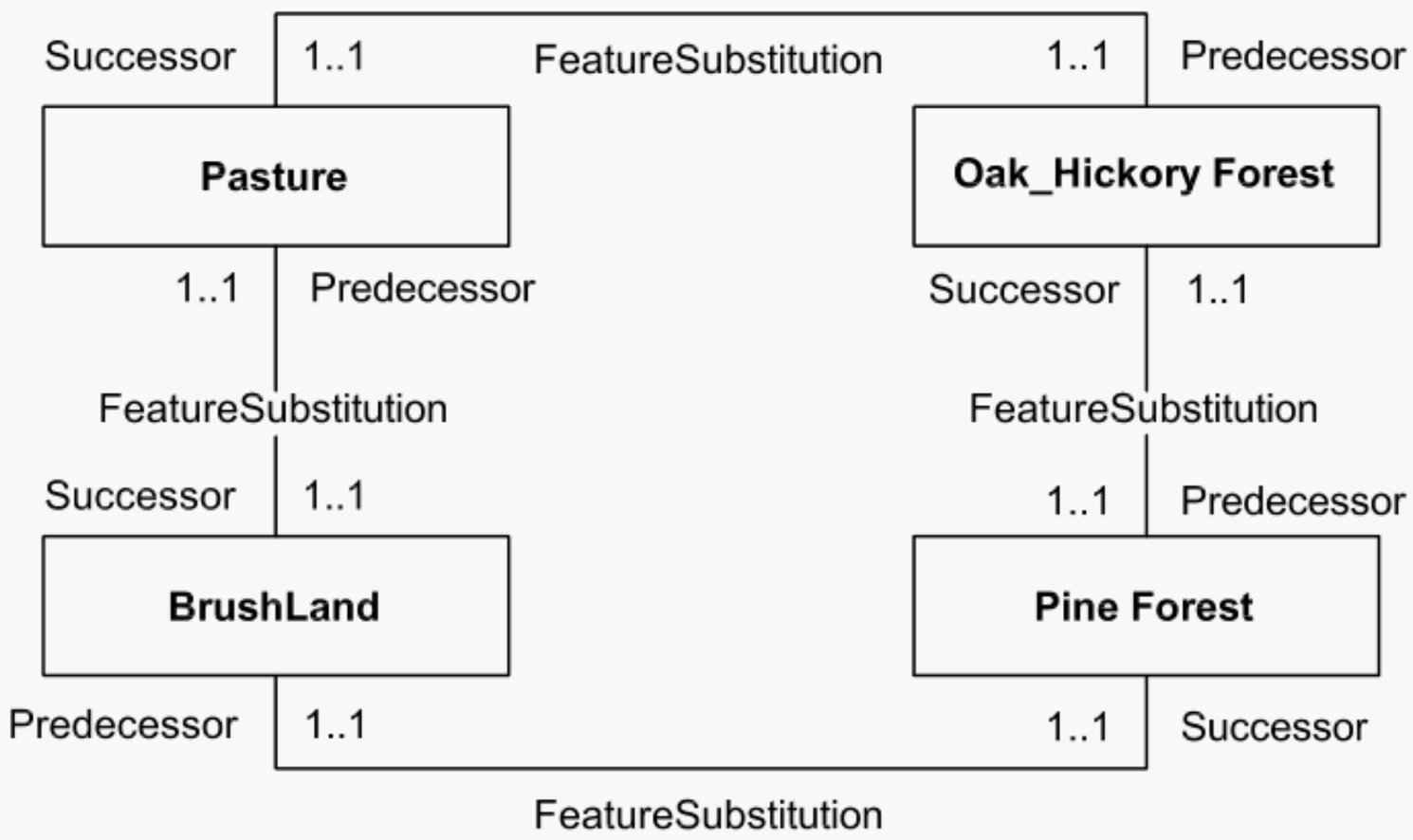

Figure B.7 — Feature succession between feature types

Figure B.8 is an example of feature succession modelled as a self-referent association of a UML class that is a supertype for the various feature types that may be involved in succession associations. Modelling in this fashion is necessary because there is no way to predict the order in which instances of these feature types might succeed each other. In this example, each of the associations has been represented as an association class, so that the time at which succession occurs can be described by an attribute of the association.

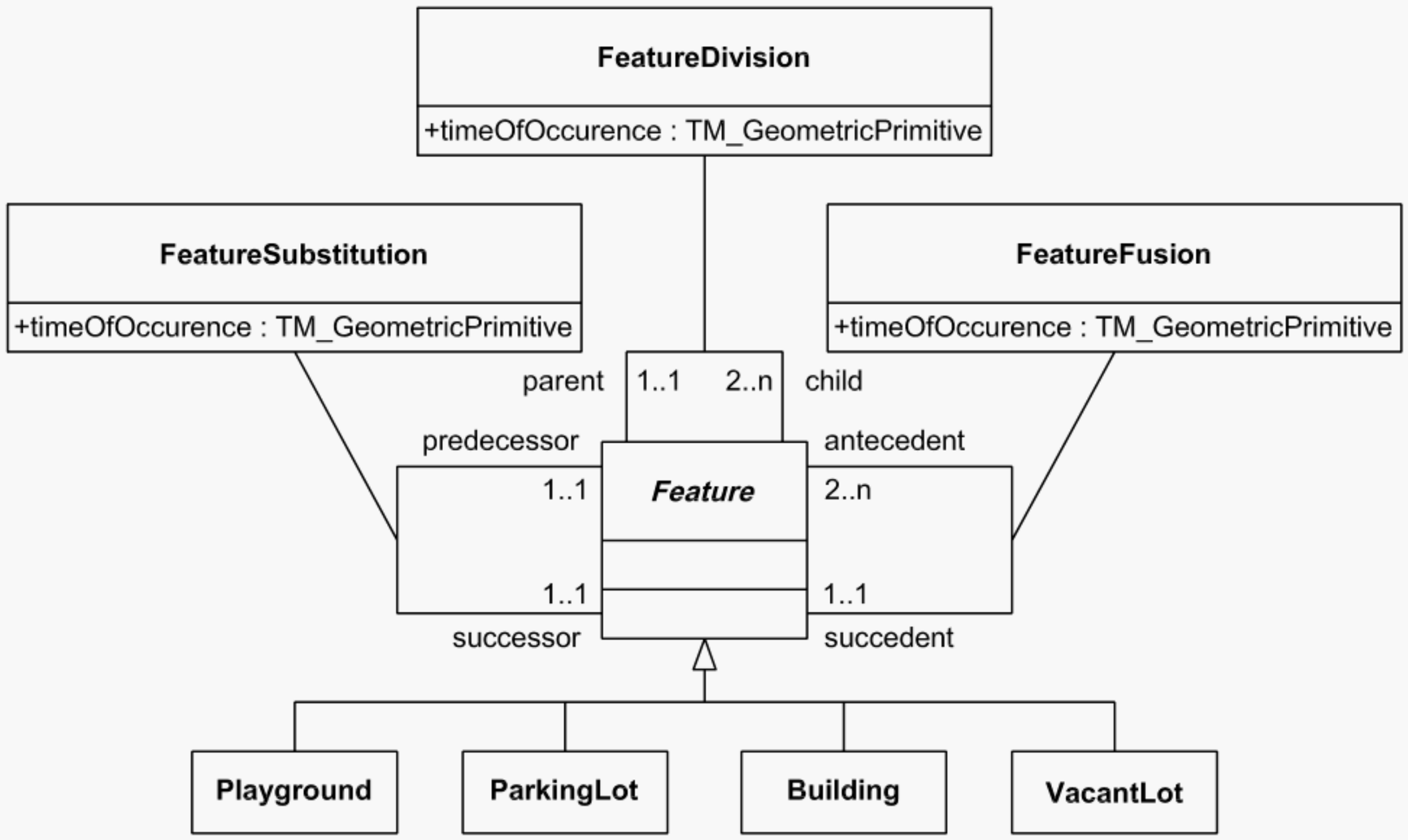

Figure B.8 — Feature succession at the generic feature level

B.3 Feature associations with temporal characteristics

Figure B.9 is an application schema for the association between a ranch headquarters and the leased rangeland that it manages. A RangelandTract is associated to a RanchHeadquarters by a Lease. A lease is an example of a feature association that would not be considered a temporal association, because time is not the dominant aspect of the association. However, a lease is in effect for a given period of time. The Lease is represented in this mode as a UML association class, which allows attributes to be assigned to an association. In this case, it has an attribute leasePeriod that identifies the period for which the lease is in effect. It could also have other attributes, such as one that identifies the cost of the lease.

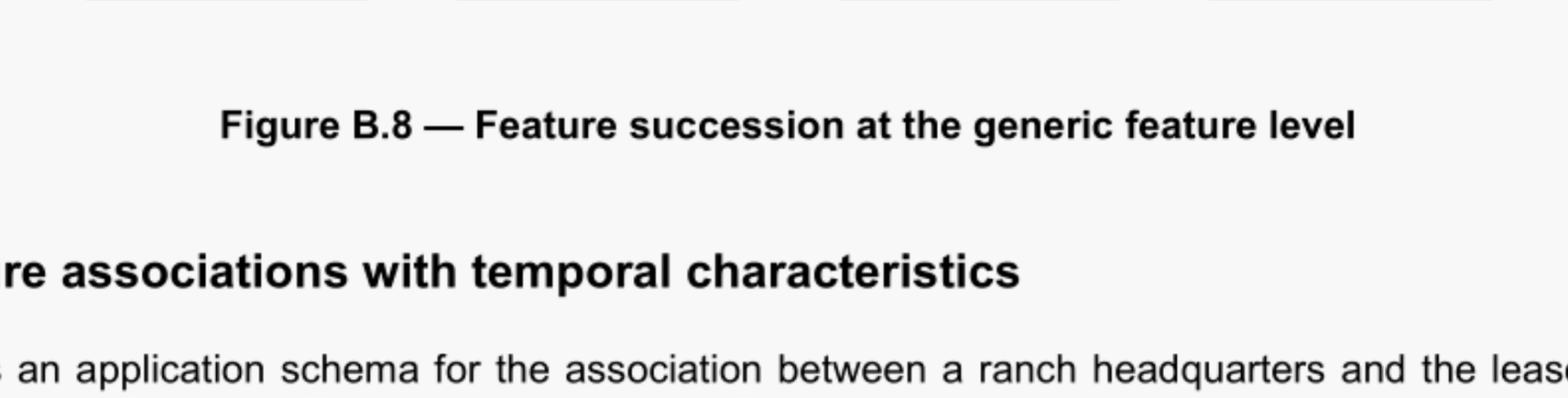

Figure B.9 — Rangeland lease

Annex C
(normative)

Describing temporal reference systems in metadata

C.1 Metadata for temporal reference systems

Subclause 5.3.1 requires that the metadata associated with a data set that uses temporal reference systems other than the Gregorian calendar and UTC shall either provide citations to documents that describe those temporal reference systems, or include description of those temporal reference systems in the metadata. The metadata elements defined in Table C.1 shall be used for this purpose. The structure of this table complies with ISO 19115:—, annex B.

Table C.1 — Metadata elements for describing temporal reference systems

	Name	Definition	Obligation/ Condition	Maximum occurrence	Model element or data type	Domain
1	TM_ReferenceSystem	Information about a temporal reference system	C/ temporal information in the data set not referenced to the Gregorian calendar?	N	class	Lines 2-33
2	name	Name by which the temporal reference system is known	M	1	RS_Identifier	ISO 19111
3	DomainOfValidity	Limits of space and time within which the temporal reference system is used	C/ Extent of temporal reference system less than extent of data set in which it is used?	N	EX_Extent	ISO/TS 19103
4	Subtype	Subtype of temporal reference system being described	M	1	Specialization	“TM_Calendar” "TM_Clock" “TM_CoordinateSystem” “TM_OrdinalReferenceSystem”
5	TM_Calendar	Description of a calendar	C/ Subtype “TM_Calendar?”	1	Class	Lines 6 – 15
6	dateTrans	Description of an operation for converting a date in the specified calendar to a Julian date	M	1	CharacterString	Free text
7	julTrans	Description of an operation for converting a Julian date to a date in the specified calendar	M	1	CharacterString	Free text
8	Basis	The calendar eras associated with the calendar being described	M	N	Association	TM_CalendarEra

Table C.1 *(continued)*

	Name	Definition	Obligation/ Condition	Maximum occurrence	Model element or data type	Domain
9	Resolution	The clock that is used with this calendar to define temporal position within a calendar day	O	1	Association	TM_Clock
10	TM_CalendarEra	Characteristics of each calendar era	M	N	Class	Lines 11-15
11	name	Name by which this calendar era is known	M	1	CharacterString	Free text
12	referenceEvent	Event used as the datum for this calendar era	M	1	CharacterString	Free text
13	referenceDate	Date of the reference event in the calendar being described	M	1	TM_CalDate	Date in the calendar being described
14	julianReference	Julian date of the reference event	M	1	JulianDate	Number
15	epochOfUse	Period for which the era was used as a basis for dating	M	1	TM_Period	ISO 8601
16	TM_Clock	Description of a clock	C/ Subtype "TM_Clock or Resolution not null?"	1	Class	Lines 17-21
17	referenceEvent	Event used as the datum for this clock	M	1	CharacterString	Free text
18	ReferenceTime	Time of the reference event for this clock	M	1	TM_ClockTime	Time in the clock being described
19	utcReference	UTC time of the reference event	M	1	TM_ClockTime	ISO 8601
20	utcTrans	Description of an operation for converting a time on this clock to a UTC time	M	1	CharacterString	Free text
21	clkTrans	Description of an operation for converting a UTC time to a time on this clock	M	1	CharacterString	Free text
22	TM_CoordinateSystem	Description of a temporal coordinate system	C/ Subtype "TM_CoordinateSystem"?	1	Class	Lines 23-26
23	origin	Position of the origin of the scale on which the temporal coordinate system is based expressed as a date in the Gregorian calendar and time of day in UTC	M	1	DateTime	ISO 8601
24	interval	Standard unit of time used to measure duration on the axis of the coordinate system	M	1	CharacterString	ISO 31-1, ISO 1000

Table C.1 *(continued)*

	Name	Definition	Obligation/ Condition	Maximum occurrence	Model element or data type	Domain
25	transformCoord	Description of an operation for converting a coordinate in this temporal coordinate system to a date in the Gregorian calendar and a time in UTC	M	1	CharacterString	Free text
26	transformDateTime	Description of an operation for converting a date in the Gregorian calendar and a time in UTC to a coordinate in this temporal coordinate system	M	1	CharacterString	Free text
27	TM_OrdinalReferenceSystem	Description of an ordinal temporal reference system	C/ Sub type “TM_OrdinalReferenceSystem?”	1	Class	Lines 28-33
28	Structure	Ordinal eras that make up the highest level of this ordinal reference system	M	1	Association	TM_OrdinalEra
29	TM_OrdinalEra	Description of an ordinal era	M	N	Class	Lines 30-33
30	name	Name that identifies a specific ordinal era	M	1	CharacterString	Free text
31	begin	Date at which the ordinal era began	O	1	DateTime	ISO 8601
32	end	Date at which the ordinal era ended	O	1	DateTime	ISO 8601
33	Composition	Ordinal eras that subdivide this ordinal era	M	1	Association	TM_OrdinalEra
NOTE The metadata element corresponding to a UML operation is a text description of the operation.						

Annex D
(informative)

Description of calendars

D.1 Internal structure of calendars

A calendar is a discrete temporal reference system that provides a basis for defining temporal position to a resolution of one day. The Gregorian Calendar is the international de facto standard. It is preferred for use with geographic information. However, there are a variety of traditional or historic calendars in addition to the Gregorian calendar. These may be appropriate for some applications of geographic information. For example, archaeological materials might be more accurately datable in the calendar of the culture under consideration. Subclause 5.3.1 requires that the metadata for any data set that uses a calendar other than the Gregorian calendar shall include a description of that calendar or a citation for a description. This annex describes some aspects of calendars that may have to be considered in such a description.

A calendar has a hierarchical structure (see Figure D.1) in which a specific type of time interval is used at each level. Typically, the instances of the intervals at one level in the hierarchy are named or numbered on the basis of a cycle whose length is equal to that of the interval used at the next higher level. Although most calendars include years and days as standard intervals, some use intervals other than months at intermediate levels. Some calendars also include additional levels within the hierarchy, or even a parallel hierarchy using other intervals. A calendar date identifies an instance of one interval at each level of the hierarchy.

EXAMPLE A date in the Gregorian calendar is identified as a specific year, a specific month within that year, and a specific day of that month.

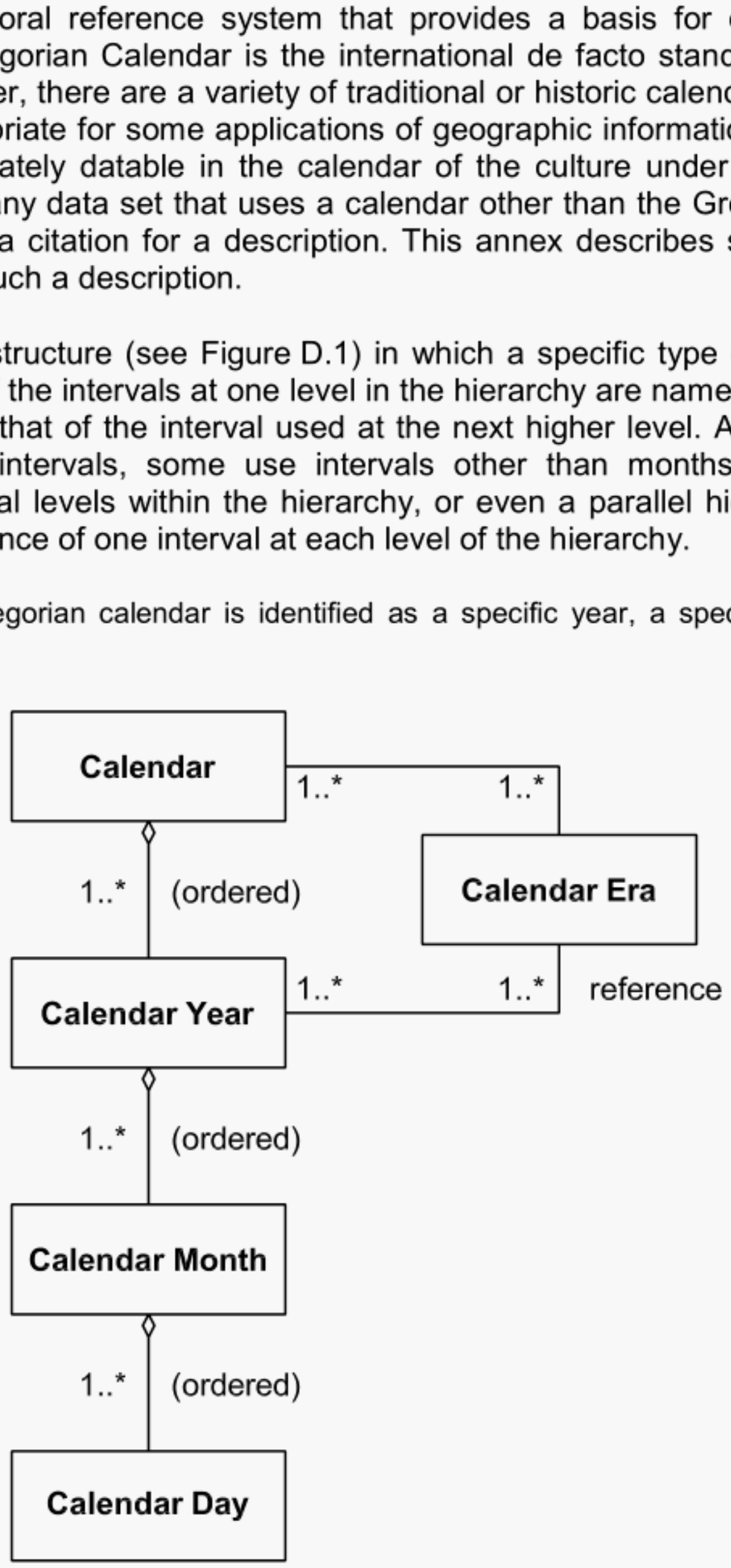

Figure D.1 — Internal structure of a typical calendar

Calendars are complex because the intervals are nominally based on astronomical cycles, but the periods of those cycles are not integer multiples of each other. In order to maintain the phase relationships between calendar cycles and astronomical cycles, the length of an interval is adjusted by intercalating intervals at lower levels.

NOTE To intercalate is to insert an additional interval, such as a day or a month, into a calendar. In the Gregorian calendar, for example, an additional day is intercalated at the end of February in each leap year.

a) The fundamental time interval of every calendar is the day, which corresponds nominally to the period of the Earth's rotation around its axis. The lengths of other intervals used in a calendar are integer multiples of a day. In many calendars, a particular day is identified by its sequence number within a month. Some calendars use other methods.

b) A calendar year corresponds nominally to the period of the Earth's revolution around the sun. Since the duration of that period is not an integer number of days, most calendars are designed to allow the length of a calendar year to vary so that the average length over a longer period equals that of the period of the Earth's revolution. Calendars vary in the precision with which this is accomplished. In most calendars, additional months or days are intercalated on a cyclical basis. In many ancient calendars, intercalation was done on an irregular basis, whenever some governmental or religious authority perceived a need to do so.

c) A year is typically composed of a sequence of months of varying lengths. When there are rules for intercalation, the internal structure of a year will follow one of a limited number of patterns each of which can be described by a template that lists the names and the lengths of the months in a year that conforms to that template. The intercalation rules provide a means for relating the sequence number of a particular year within an era to the template to which it conforms.

   A month corresponds nominally to the period of a lunar cycle — either the cycle of the phases of the moon, or that of the moon's revolution around the Earth. In many ancient calendars, the length of a month was determined by astronomical observation rather than by computation. Neither lunar cycle has a period equal to an integer multiple of the length of the day, nor is an astronomical year an integer multiple of the period of either lunar cycle. Some calendars maintain the phase relationship between the calendar month and the lunar cycle by restricting the variation in the length of the calendar month in such a way that its average length equals the period of the lunar cycle. In this case, the calendar year is readjusted to the period of the Earth's revolution around the sun by intercalating an additional month in specified years. Other calendars — the Gregorian Calendar, for example — are not designed to be kept in phase with a lunar cycle: the calendar year is simply divided into an integer number of months.

d) Many calendars include an interval of several days that is shorter than a month. The Gregorian and other western calendars, for example, use a seven day week as a standard interval. Although the day of the week may be a culturally significant element of the date, it adds no information about the temporal position of the day. Intercalation rules may be complicated because of requirements that certain festivals fall on particular days of the week as well as on specific dates.

e) Every calendar is associated with one or more calendar eras. A calendar era is a sequence of years counted relative to a reference date that corresponds to some mythical or historic event. Calendars tend to fall into two groups. Some count years relative to a single reference event, so that years fall into only one or two calendar eras. The Gregorian calendar is an example. For a variety of reasons, the reference event may change from one historical period to another. The Julian calendar (D.3.1), for example, has been used with several different reference events. Other calendars use multiple reference events. Regnal dating — the counting of years from ascent of a ruler — is quite common. The modern Japanese calendar (D.3.2) and the ancient Babylonian calendar (D.3.3.) are examples.

D.2 Describing a calendar

Subclause 5.3.2 describes a schema that defines a set of relatively simple interfaces for a calendar. It specifies detailed requirements for the description of a calendar era. The nature of the information that can be provided to support transformation of calendar dates to Julian dates are conditioned by some of the factors discussed in D.1.

Some calendars, including most of those in current use, have been regularized to such an extent that algorithms can be written to convert a date precisely to a Julian date. In this case, the required information may be provided in

the form of such an algorithm. Dershowitz and Reingold (1997), Doggett (1992), Hatcher (1984, 1985), and Richards (1998) provide a number of such algorithms.

Many calendars are nearly regular; algorithms can be written to convert dates to approximate values for Julian dates. In this case, the required information may be provided in the form of such algorithms. It should include an estimate of the accuracy of the result for each algorithm, and a statement of the period within which such accuracy can be attained.

Very irregular calendars cannot be described adequately by algorithms. It may be possible in some cases to provide a description of each of the annual patterns that occurs together with a look up table that identifies the pattern used in each year. It may only be possible to provide a description of a typical calendar year with a set of reference dates that would allow dates to be transformed to approximate Julian dates by interpolation. Parise (1982) provides numerous calendar conversion tables.

D.3 Examples

D.3.1 Julian calendar

The Julian calendar is an example of a regular calendar. It dates from 45 BC, when Julius Caesar imposed a reform of the ancient Roman calendar in order to bring it back into synchrony with the solar year. The common year of the Julian calendar has a duration of 365 days, with an additional day intercalated in every fourth year, which is known as a leap year. After Caesar's death, the Roman pontifices misinterpreted the rule, and intercalated a day in every third year. In 9 BC, Augustus ordered that the resulting error be corrected by omitting the intercalation in each of the succeeding leap years until 8 AD.

The Julian calendar system has been used with a number of calendar eras. See Parise (1982) and Richards (1998) for details. During the late republican era through the imperial age of Rome, years were counted from the supposed date of the founding of the city, equivalent to 753 BC. In 525 AD, Dionysius Exiguus proposed numbering years from the supposed birth of Jesus Christ. However, this Christian era was not widely used in Western Europe until the 11th century AD, and was not adopted in the Greek world until the 15th century.

Table D.1 — Months of the Julian year

Month Name	Length in common year	Length in leap year
January	31	31
February	28	29
March	31	31
April	30	30
May	31	31
June	30	30
July	31	31
August	31	31
September	30	30
October	31	31
November	30	30
December	31	31

Since the time of Augustus, the Julian year has been composed of 12 months, as shown in Table D.1. In the Roman Empire, January 1 was the first day of the year. In later times, other dates were used. See Parise (1982) for details.

Table D.2 is an example of a description of the Julian calendar that satisfies the requirements of 5.3. The example describes the Julian calendar as used with the Christian era from about 1 January 1000 AD until 4 October 1582 AD, with 1 January as the first day of the year.

Table D.2 — Description of the Julian calendar with the Christian era

Element	Value
TM_ReferenceSystem	
TM_ReferenceSystem.name	Julian calendar
TM_ReferenceSystem.domainOfValidity	Western Europe
TM_CalendardateTrans Algorithm for transformation of Julian calendar date (Y/M/D) to Julian date (J).	$Y' = Y + 4716 - (14 - M) / 12$ $M' = MOD(M + 9, 12)$ $D' = D - 1$ $c = (1461Y') / 4$ $d = (153M' + 2) / 5$ $J = c + d + D' - 1401$
TM_Calendar.julTrans Algorithm for transformation of Julian date (J) to Julian calendar date (Y/M/D).	$J' = J + 1401$ $Y' = (4J' + 3) / 1461$ $T' = MOD(4J' + 3, 1461) / 4$ $M' = (5T' + 2) / 153$ $D' = MOD(5T' + 2, 153) / 5$ $D = D' + 1$ $M = MOD(M' + 2, 12) + 1$ $Y = Y' - 4716 + (14 - M) / 12$
Basis	
TM_CalendarEra name	Christian era
TM_CalendarEra.referenceEvent	Birth of Jesus Christ
TM_CalendarEra.referenceDate	01, 01, 01
TM_CalendarEra.julianReference	1721424
TM_CalendarEra.epochOfUse.begin	2087769
TM_CalendarEra.epochOfUse.end	2299160

The algorithms in Table D.2 have been adapted from Richards (1998). They need to be modified to support transformations of Julian calendar dates associated with other calendar eras, or for Julian calendars that use a date other than January 1 for the first day of the year. See Richards (1998) for details. All calculations are done in integer arithmetic.

D.3.2 Modern Japanese calendar

Since 1873, Japan has used the Gregorian calendar, except that the system of numbering years according to regnal eras has been maintained. Table D.3 exemplifies the use of the elements specified by this International Standard for describing calendar eras.

Table D.3 — Japanese calendar eras

name	referenceEvent	referenceDate	julianReference	epochOfUse	
				begin	end
Meiji	adoption of Gregorian calendar	Meiji 6.1.1	2405160	2405160	2419614
Taisho	New emperor accession	Taisho 1.7.31	2419615	2419615	2424875
Showa	New emperor accession	Showa 1.12.26	2424876	2424876	2447534
Heisei	New emperor accession	Heisei 1.1.8	2447535	2447535	current

D.3.3 Ancient Babylonian calendar

The ancient Babylonian calendar is an example of a very irregular calendar. It was a lunar calendar, with twelve months in the typical year (Table D.4). A thirteenth month was intercalated to keep the calendar aligned with the seasons. At first (ca. 2400 BC) this was done as needed to ensure that the barley harvest occurred during the first month of the year. By 1000 BC, the decision to intercalate depended upon the proximity of the moon to the Pleiades on the first day of the year. By 500 BC, intercalation was done on a 19 year cycle, adding an Adaru II in years 3, 6, 8, 11, 14, and 19, and an Ululu II in year 17.

Table D.4 — Months of the Babylonian year

Number	Name	Number	Name
1	Nisanu	7	Tashritu
2	Ayaru	8	Arakhsamnu
3	Simanu	9	Kislimu
4	Du'uzu	10	Tebetu
5	Abu	11	Shabatu
6	Ululu	12	Adaru
(6)	Ululu II	(12)	Adaru II

The day began at sunset. Normally the month began when the new moon was first seen, so the length of the months alternated irregularly between 29 and 30 days. However, the month ended on the 30th day if the new moon could not be observed.

NOTE 1 In most ancient calendars, the day began either at sunrise or at sunset, because they are the only two events of the diurnal cycle that can be observed without the aid of specialized equipment. Starting the day at sunset is consistent with starting the month when the new moon is first observed.

NOTE 2 In principle, it is possible to calculate the length of each lunar month from modern astronomical data, but, because the lengths of the calendar months were based on observations, they may not have corresponded to the astronomical months.

Years were numbered on the basis of regnal eras. The first year of an era was the first to begin after the king's ascent to the throne. In the second century AD, Ptolemy produced a list of kings and their dates, going back to 747 BC. This list has been used as a basis for correlating Babylonian years with years in more recent eras. Other ancient lists have been found, which have allowed the dating to be extended back to about 2000 BC, although with less accuracy for the earlier years. Table D.5 provides an example for ten years of the Babylonian calendar.

Table D.5 — First 10 Years of the Reign of Nebuchadnezzar II in Babylon

Year	Julian date for 1 Nisanu	Number of months	Number of days	Intercalated month
Nebuchadnezzar 1	1500163.25	13	384	Adaru II
Nebuchadnezzar 2	1500547.25	12	355	
Nebuchadnezzar 3	1500902.25	12	356	
Nebuchadnezzar 4	1501258.25	13	382	Ululu II
Nebuchadnezzar 5	1501640.25	12	355	
Nebuchadnezzar 6	1501995.25	12	354	
Nebuchadnezzar 7	1502349.25	13	383	Adaru II
Nebuchadnezzar 8	1502732.25	12	355	
Nebuchadnezzar 9	1503085.25	13	383	Adaru II
Nebuchadnezzar 10	1503470.25	12	355	

Because the lengths of particular months are not known, it is impossible to perform an exact transformation of a date in this calendar to a Julian date. However, an algorithm based on the average length of a month in any given year will return a close approximation of the correct Julian date. Table D.6 exemplifies the use of the elements specified by this International Standard for describing the Babylonian calendar.

Table D.6 — Description of the Babylonian calendar from the time of Nebuchadnezzar II

Element	Value
TM_ReferenceSystem	
TM_ReferenceSystem.name	Babylonian calendar
TM_ReferenceSystem.domainOfValidity	Southwest Asia
TM_Calendar.dateTrans Algorithm for transformation of Babylonian calendar date (Y/M/D) to Julian date (J)	$J = J_Y + INT((M-1)(L/N)) + D - 1$ where: J_Y – Julian date for the 1st day of Nisanu for year Y (from table) M – ordinal number of the month L – number of days in the year (from table) N – number of months in the year (from table) D – ordinal number of the day within the month
TM_Calendar.julTrans Algorithm for transformation of Julian date (J) to Babylonian calendar date	Y = value from table for the year for which the Julian date of 1 Nisanu (J_Y) is the largest value less than J $M = INT((J - J_Y + 1)/(L/N))$ $D = J - J_Y + 1 - INT(M*(L/M))$
Basis	
TM_Calendar era name	Nebuchadnezzar II
TM_CalendarEra.referenceEvent	Ascension of Nebuchadnezzar II to the throne of Babylon
TM_CalendarEra.referenceDate	01, 01, 01
TM_CalendarEra.JulianReference	1721423.25
TM_CalendarEra.epochOfUse.begin	2087769
TM_CalendarEra.epochOfUse.end	2299160

D.3.4 Global Positioning System calendar

The Global Positioning System (GPS) uses a system specific calendar and clock for stating the temporal position of GPS related data. Dates are identified in terms of a week number (WN) counted within a calendar era that began at midnight (00:00:00 UTC) on 6 January 1980, and days are identified by the ordinal number of the day (DN) within a week. In both cases, numbering starts with zero. Table D.7 exemplifies the use of the elements specified by this International Standard for describing the GPS calendar.

Table D.7 — Description of the GPS calendar

Element	Value
TM_ReferenceSystem	
TM_ReferenceSystem.name	Global Positioning System calendar
TM_ReferenceSystem.domainOfValidity	global
TM_Calendar.dateTrans Algorithm for transformation of calendar date (WN, DN) to Julian date (J)	J = 2444244.5 + (7 * WN) + DN
TM_Calendar.julTrans Algorithm for transformation of Julian date (J) to calendar date (WN.DN)	DL = J – 2444244.5 WN = INT (DL/7) DN = MOD (DL, 7)
Basis	
TM_CalendarEra.name	GPS era
TM_CalendarEra.referenceEvent	
TM_CalendarEra.referenceDate	0001, 01
TM_CalendarEra.julianReference	2444244.5
TM_CalendarEra.epochOfUse.begin	2444244.5
TM_CalendarEra.epochOfUse.end	current

The GPS time system is unusual in that it specifies clock time as time of week (TOW) expressed in seconds, rather than time of day. A GPS week consists of 604,800 seconds. GPS time differs from UTC for two reasons. GPS time is a continuous scale, whereas UTC is periodically readjusted to the periodic time of the Earth's rotation by the addition of leap seconds. GPS time also drifts within a one microsecond range relative to UTC. As a result, the algorithms for converting between GPS time and UTC are rather complicated and are not included in this example.

Bibliography

[1] ALLEN, J. F., *Maintaining Knowledge about Temporal Intervals,* Communications of the ACM, 1983, vol. 26 pp. 832-843

[2] N. DERSHOWITZ and E. M. REINGOLD, *Calendrical calculations,* Cambridge University Press, 1997

[3] L. E. DOGGETT, *Calendars* in P. K. SEIDELMANN (editor), Explanatory supplement to the astronomical almanac and the American ephemeris and nautical almanac, University Science Books, Sausalito, CA, USA, 1992, pp. 575-608

[4] D. A. HATCHER, *Simple formulae for Julian day numbers and calendar dates,* Journal of the Royal Astronomical Society, 1984, Vol. 25, p. 53

[5] D. A. HATCHER, *Generalized equations for Julian day numbers and calendar dates.* Journal of the Royal Astronomical Society, 1985, Vol. 26, p. 151

[6] International Telecommunications Union, ITU -R Recommendation TF.686-1 (10/97), Glossary, 1997

[7] International Telecommunications Union, ITU-R Recommendation TF.460-5 (10/97), Standard-frequency and time-signal emissions, 1997

[8] ISO 31-2:1992, *Quantities and units — Part 2: Periodic and related phenomena*

[9] ISO 19104:—[2)], *Geographic information — Terminology*

[10] C. S. JENSEN, et al. *A consensus glossary of temporal data base concepts,* ACM SIGMOD Records, 1994, Vol. 23 Also available as consGlos.ps from ftp://ftp.cs.arizona.edu/tsql/doc/

[11] Object Management Group, *OMG Unified Modeling Language Specification, version 1.*3 1999, Available from World Wide Web at http://www.omg.org/cgi-bin/doc?ad/99-06-08

[12] F. PARISE, *The book of calendars*, Facts on file, New York, 1982

[13] J. P.PARISOT, *Additif to the paper of H. A. Hatcher,* Journal of the Royal Astronomical Society, 1986, *Vol. 27, p. 506*

[14] E. G. RICHARDS, *Mapping Time: The calendar and its history*. Oxford University Press, 1998

2) To be published.

ICS 35.240.70

Price based on 48 pages