Open Geospatial Consortium

OGC Dynamic Features Discussion Paper

Copyright notice

Warning

License Agreement

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD.

THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications. This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

Table of Contents

i. Abstract

Moving Feature standards and technologies have made considerable progress. It is time to evaluate the foundations of OGC standards to determine if they are sufficient to support current and future Moving Feature standards. This discussion paper examines ISO and OGC standards against current and anticipated future needs. It also examines other standards and conventions which may be useful for enhancing the OGC/ISO foundation.>

ii. Keywords

The following are keywords to be used by search engines and document catalogues.

ogcdoc, OGC document, <tags separated by commas>

iii. Preface

NOTE	Insert Preface Text here. Give OGC specific commentary: describe the technical content, reason for document, history of the document and precursors, and plans for future work. > Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights. Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

iv. Submitting organizations

The following organizations submitted this Document to the Open Geospatial Consortium (OGC):

Organization name(s)

v. Submitters

All questions regarding this submission should be directed to the editor or the submitters:

Name Affiliation

Chapter 1. Scope

Moving Feature standards and technologies have made considerable progress. It is time to evaluate the foundations of OGC standards to determine if they are sufficient to support current and future Moving Feature standards. This discussion paper examines ISO and OGC standards against current and anticipated future needs. It also examines other standards and conventions which may be useful for enhancing the OGC/ISO foundation.

Chapter 2. References

The following normative documents contain provisions that, through reference in this text, constitute provisions of this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

NOTE

Insert References here. If there are no references, state "There are no normative references".

References are to follow the Springer LNCS style, with the exception that optional information may be appended to references: DOIs are added after the date and web resource references may include an access date at the end of the reference in parentheses. See examples from Springer and OGC below.

Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195–197 (1981)

May, P., Ehrlich, H.C., Steinke, T.: ZIB Structure Prediction Pipeline: Composing a Complex Biological Workflow through Web Services. In: Nagel, W.E., Walter, W.V., Lehner, W. (eds.) Euro-Par 2006. LNCS, vol. 4128, pp. 1148–1158. Springer, Heidelberg (2006)

Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)

Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid Information Services for Distributed Resource Sharing. In: 10th IEEE International Symposium on High Performance Distributed Computing, pp. 181–184. IEEE Press, New York (2001)

Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration. Technical report, Global Grid Forum (2002)

National Center for Biotechnology Information, http://www.ncbi.nlm.nih.gov

ISO / TC 211: ISO 19115-1:2014 Geographic information — Metadata — Part 1: Fundamentals (2014)

ISO / TC 211: ISO 19157:2013 Geographic information — Data quality (2013)

ISO / TC 211: ISO 19139:2007 Geographic information — Metadata — XML schema implementation (2007)

ISO / TC 211: ISO 19115-3: Geographic information — Metadata — Part 3: XML schemas (2016)

OGC: OGC 15-097 OGC Geospatial User Feedback Standard. Conceptual Model (2016)

OGC: OGC 12-019, OGC City Geography Markup Language (CityGML) Encoding Standard (2012)

OGC: OGC 14-005r3, OGC IndoorGML (2014)

Chapter 3. Terms and Definitions

This document uses the terms defined in Sub-clause 5.3 of [OGC 06-121r8], which is based on the ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards. In particular, the word "shall" (not "must") is the verb form used to indicate a requirement to be strictly followed to conform to this Best Practice.

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

3.1. base representation

<moving features representation, using a local origin and local ordinate vectors, of a geometric object at a given reference time [ISO 19141:2008, definition 4.1.1]

NOTE 1 A rigid geometric object may undergo translation or rotation, but remains congruent with its base representation.

NOTE 2 The local origin and ordinate vectors establish an engineering coordinate reference system (ISO 19111), also called a local frame or a local Euclidean coordinate system.

3.2. design coordinate reference system

engineering coordinate reference system in which the base representation of a moving object is specified. [ISO 19141:2008, definition 4.1.3]

3.3. direct position

DirectPosition object data types hold the coordinates for a position within some coordinate reference system.

3.4. external coordinate reference system

the coordinate reference system in which the geometry of the curve for a trajectory is defined. Usually an earth-fixed geographic CRS. [ISO 19141:2008, derived]

3.5. feature

abstraction of real world phenomena [ISO 19101:2002, definition 4.11]

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

3.6. foliation

one parameter set of geometries such that each point in the prism of the set is in one and only one trajectory and in one and only one leaf [ISO 19141:2008, definition 4.1.8]

3.7. geometric object

spatial object representing a geometric set [ISO 19107:2003, definition 4.47]

3.8. geometric primitive

geometric object representing a single, connected, homogeneous element of space [ISO 19107:2003, definition 4.48]

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

3.9. global coordinate system

the CRS of the trajectory of the reference point.

NOTE A global coordinate system is usually in terms of the moving frame of the curve (oriented on the local tangent) or in terms of the external CRS in which the geometry of the curve is defined. [ISO 19141:2008, derived]

3.10. GM_Point

GM_Point is the basic data type for a geometric object consisting of one and only one point. That point is represented as a DirectPosition.

3.11. instant

0-dimensional geometric primitive representing position in time [ISO 19108:2002, definition 4.1.17]

3.12. leaf

<one parameter set of geometries> geometry at a particular value of the parameter [ISO 19141:2008, definition 4.1.12]

3.13. one parameter set of geometries

function f from an interval t ⏥ [a, b] such that f(t) is a geometry and for each point P ⏥ f(a) there is a one parameter set of points (called the trajectory of P) P(t) : [a, b] → P(t) such that P(t) ⏥ f(t) [ISO 19141:2008, definition 4.1.15]

EXAMPLE A curve C with constructive parameter t is a one parameter set of points c(t).

3.14. prism

<one parameter set of geometries> set of points in the union of the geometries (or the union of the trajectories) of a one parameter set of geometries [ISO 19141:2008, definition 4.1.18]

NOTE This is a generalization of the concept of a geometric prism that is the convex hull of two congruent polygons in 3D-space. Such polyhedrons can be viewed as a foliation of congruent polygons.

3.15. 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 [ISO 19108:2002, definition 4.1.31]

3.16. temporal position

location relative to a temporal reference system [ISO 19108:2002, definition 4.1.34]

3.17. temporal reference system

reference system against which time is measured [ISO 19108:2002, definition 4.1.35]

3.18. trajectory

path of a moving point described by a one parameter set of points [ISO 19141:2008, definition 4.1.22]

3.19. vector

quantity having direction as well as magnitude [ISO 19123:2005, definition 4.1.43]

Chapter 4. Conventions

This sections provides details and examples for any conventions used in the document. Examples of conventions are symbols, abbreviations, use of XML schema, or special notes regarding how to read the document.

4.1. Identifiers

The normative provisions in this document are denoted by the URI

http://www.opengis.net/spec/{standard}/{m.n}

All requirements and conformance tests that appear in this document are denoted by partial URIs which are relative to this base.

NOTE	A description of UML and how it is used in ISO and OGC standards will be included here.

Chapter 5. Dynamic Features

Introductory text

5.1. Geometry in 3 Dimensions

Topic: Is the OGC geometry model sufficient to support 3D?

First we need to define 3D. Our proposed definition is as follows:

3D Feature: A Feature whos' geometry is defined by a volume enclosed by a boundary surface.

This definition is consistent with ISO 19107. This ISO standard defines both the GM_Object and GM_Boundary classes as root level geometry classes. The association between object and boundary is achieved through the "boundary()" operation on the GM_Object class.

class Geometry

«type»
Geometry root::GM_Object

+ boundary(): GM_Boundary
+ buffer(Distance*): GM_Object
+ centroid(): DirectPosition
+ closure(): GM_Complex
+ convexHull(): GM_Object
+ coordinateDimension(): Integer
+ dimension(DirectPosition*): Integer
+ distance(GM_Object*): Distance
+ envelope(): GM_Envelope
+ isCycle(): Boolean
+ isSimple(): Boolean
+ maximalComplex(): Set<GM_Complex>
+ mbRegion(): GM_Object
+ representativePoint(): DirectPosition
+ transform(SC_CRS*): GM_Object

The GM_Object "boundary()" operation returns a GM_Boundary.

"Any subclass of GM_Object will use a subclass of GM_Boundary to represent its boundary through the operation GM_Object::boundary."

«Abstract»
Geometry root::GM_Boundary

+ boundary(): GM_Boundary
+ buffer(Distance*): GM_Object
+ centroid(): DirectPosition
+ closure(): GM_Complex
+ convexHull(): GM_Object
+ coordinateDimension(): Integer
+ dimension(DirectPosition*): Integer
+ distance(GM_Object*): Distance
+ envelope(): GM_Envelope
+ isCycle(): Boolean
+ isMaximal(): Boolean
+ isSimple(): Boolean
+ maximalComplex(): Set<GM_Complex>
+ mbRegion(): GM_Object
+ representativePoint(): DirectPosition
+ transform(SC_CRS*): GM_Object

«type»
Geometric primitive::GM_Primitive

+ boundary(): GM_PrimitiveBoundary
+ GM_Primitive(GM_Envelope*): GM_Primitive

+containingPrimitive 0..*

Interior to

+containedPrimitive 0..*

«Abstract»
Geometric primitive:: GM_PrimitiveBoundary

«type»
Geometric primitive::GM_Solid

+ area(): Area
+ boundary(): GM_SolidBoundary
+ GM_Solid(GM_SolidBoundary*): GM_Solid
+ volume(): Volume

The boundary of GM_Solids shall be represented as GM_SolidBoundary.

«type»
Geometric primitive:: GM_SolidBoundary

1   1

+interior   0..*   +exterior   0..1

Measure
Units of Measure::Volume

+ uom: UomVolume

Measure
Units of Measure::Area

+ uom: UomArea

GM_CompositeSurface
«type»
Geometric primitive:: GM_Shell

GM_Object and GM_Boundary are dimensionless. Further refinement is needed to fully define a 3D geometry.

5.1.1. Volumetrics

Starting with volumetrics, GM_Object is subclassed into GM_Primitive then into GM_Solid. The "volume()" operation on GM_Solid returns the volume (defined in ISO 19103) associated with a Feature. Thus, ISO 19107 supports the concept of a 3D volume.

Real 3D objects are often not solid. So the 3D model must also support voids, or even entire 3D Features within their interior. GM_Primitive addresses this need through the "interior to" association. The two roles on this association are the containingPrimitive (the GM_Primitive which contains another GM_Primitive) and the containedPrimitive (the GM_Primitive which is contained). This association has proven its worth in 2D space so there is little doubt that it will be just as effective in 3D.

5.1.2. Boundaries

A 3D volume is delineated by a bounding surface. GM_Boundary is the root class for boundaries. The subclass GM_PrimitiveBoundary provides the boundary for GM_Primitives. The GM_PrimitiveBoundary subclass GM_SolidBoundary is defined as the boundary for a GM_Solid.

class Boundary

«type»
Geometric
primitive::GM_Shell

GM_Shell is the exterior
surface, or boundary, of
the contained volume.

GM_Composite
«type»
Geometric complex::
GM_CompositeSurface

+composite 0..*

Composition

+generator 1..*

GM_OrientablePrimitive
«type»
Geometric primitive::
GM_OrientableSurface

+ boundary(): GM_SurfaceBoundary

A GM_Shell is a GM_CompositeSurface
A GM_CompositeSurface is composed of GM_OrientablePrimitives
A GM_Surface is a type of GM_OrientablePrimitive
A GM_PolyhedralSurface is a type of GM_Surface
A GM_PolyhedralSurface can be composed of GM_Polygons

«type»
Geometric primitive::GM_Surface

+ GM_Surface(GM_SurfaceBoundary*): GM_Surface
+ GM_Surface(GM_SurfacePatch*): GM_Surface

+surface 0..1

Segmentation

+patch 1..*

«Abstract»
Geometric primitive::GM_SurfacePatch
{root}

+ interpolation: GM_SurfaceInterpolation = "planar"
+ numDerivativesOnBoundary: Integer [0..1] = 0

+ boundary(): GM_SurfaceBoundary

«type»
Coordinate geometry::GM_PolyhedralSurface

+ GM_PolyhedralSurface(GM_Polygon*): GM_PolyhedralSurface

+surface 0..1

Segmentation

+patch 1..*

«type»
Coordinate geometry::GM_Polygon

+ boundary: GM_SurfaceBoundary
+ spanningSurface: GM_Surface [0..1]

+ GM_Polygon(GM_SurfaceBoundary*, GM_Surface*): GM_Polygon
+ GM_Polygon(GM_SurfaceBoundary*): GM_Polygon

ISO 19107 goes even farther. A GM_SolidBoundary is composed of both interior and exterior boundaries. These boundaries are defined by the GM_Shell class. Following the class associations we see:

- A GM_Shell is a GM_CompositeSurface

- A GM_CompositeSurface is composed of GM_OrientablePrimitives

- A GM_Surface is a type of GM_OrientablePrimitive

- A GM_PolyhedralSurface is a type of GM_Surface

- A GM_PolyhedralSurface can be composed of GM_Polygons

A surface constructed of polygons is an example of Boundary Representation (B-Rep) of a surface. This approach is fundamental to rendering 3D computer graphics. (ref Adam Powers 1981)

5.1.3. Closure Surfaces

Some structures, such as a tunnel or overpass, pose difficulties for this geometry model. The boundary surface can be constructed so that it continues into the interior of the structure. That would make the interior of a tunnel external to the tunnel object. This is not always a desireable result. CityGML provides the concept of a "Closure Surface".

A Closure Surface is a surface which is a logical part of the object but does not correspond to a physical part of the object. For example, the entrance to a tunnel can have a closure surface. This surface allows you to treat the tunnel as a single three-dimension entity, even though there is a hole in the bounding surface.
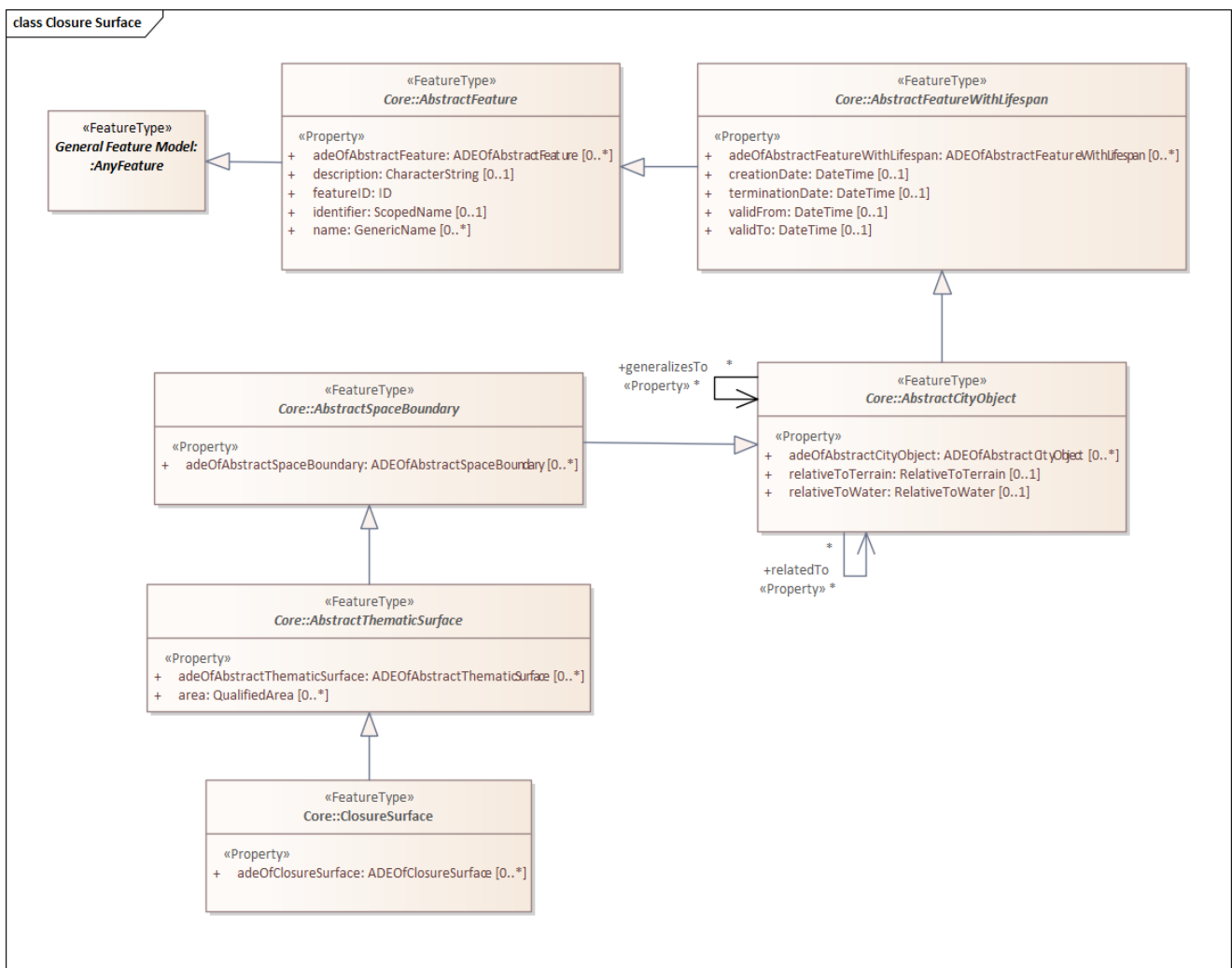
As implemented in CityGML 3.0, the ClosureSurface class has quite an ancestory. We may want to generalize this concept for use outside of CityGML. However, the capabilties provided by the ancestor classes do provide value and may be worth incorporating into a general 3D model.

5.1.4. B-Rep

The polyhedral surfaces which bound volumetric shapes are similar to the Boundary Representation (B-Rep) approach used in CAD and computer graphics. B-Rep defines a 3-dimensional surface which serves as the interface between the interior of the volumetric shape and the exterior. This surface is usually defined by a collection of shape elements (polygons) which together form a closed surface.

https://en.wikipedia.org/wiki/Boundary_representation

5.1.5. Point Clouds

"In addition to the spatial representations defined in the Core module, the geometry of physical spaces and of thematic surfaces can now also be provided by 3D point clouds using MultiPoint geometry. This allows, for example, spatially representing the building hull, a room within a building or a single wall surface just by a point cloud. All thematic feature types including transportation objects, vegetation, city furniture, etc. can also be spatially represented by point clouds. In this way, the ClearanceSpace of a road or railway could, for instance, be modelled directly from the result of a mobile laser scanning campaign. Point clouds can either be included in a CityGML dataset or just reference an external file of some common types such as LAS or LAZ."

5.1.6. SPICE

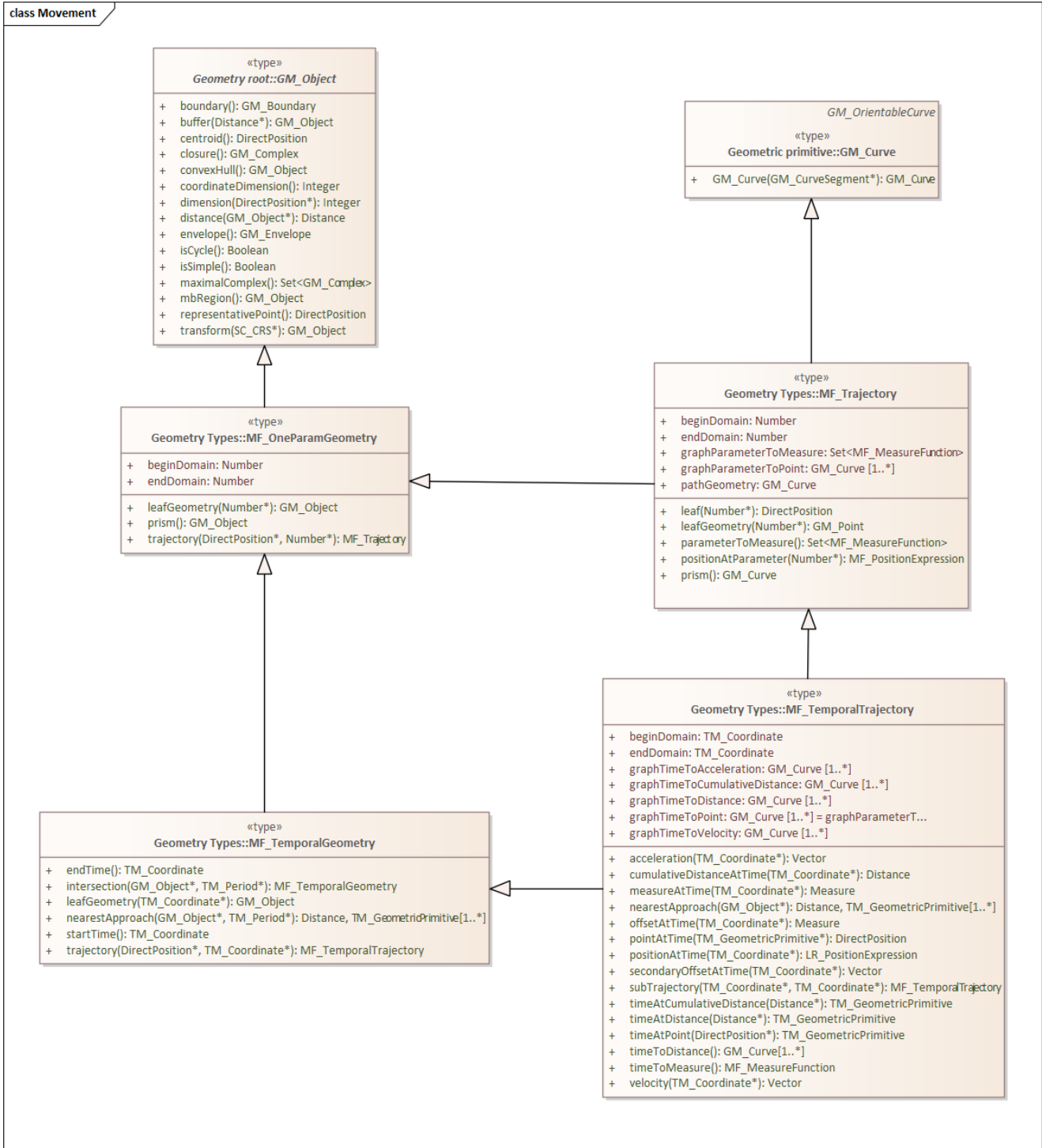https://naif.jpl.nasa.gov/naif/toolkit.html

This section will contain a discussion of SPICE and how well it aligns with the ISO/OGC concepts.

5.2. Moving Features

Topic: Is the OGC geometry model sufficient to support 3D Moving Features?

ISO 19141 is the standard for Moving Features. It introduces Feature Properties whose values are a function of an input parameter. Typically that parameter is time.

class Movement

«type»
Geometry root::GM_Object

+ boundary(): GM_Boundary
+ buffer(Distance*): GM_Object
+ centroid(): DirectPosition
+ closure(): GM_Complex
+ convexHull(): GM_Object
+ coordinateDimension(): Integer
+ dimension(DirectPosition*): Integer
+ distance(GM_Object*): Distance
+ envelope(): GM_Envelope
+ isCycle(): Boolean
+ isSimple(): Boolean
+ maximalComplex(): Set<GM_Complex>
+ mbRegion(): GM_Object
+ representativePoint(): DirectPosition
+ transform(SC_CRS*): GM_Object

GM_OrientableCurve

«type»
Geometric primitive::GM_Curve

+ GM_Curve(GM_CurveSegment*): GM_Curve

«type»
Geometry Types::MF_OneParamGeometry

+ beginDomain: Number
+ endDomain: Number

+ leafGeometry(Number*): GM_Object
+ prism(): GM_Object
+ trajectory(DirectPosition*, Number*): MF_Trajectory

«type»
Geometry Types::MF_Trajectory

+ beginDomain: Number
+ endDomain: Number
+ graphParameterToMeasure: Set<MF_MeasureFunction>
+ graphParameterToPoint: GM_Curve [1..*]
+ pathGeometry: GM_Curve

+ leaf(Number*): DirectPosition
+ leafGeometry(Number*): GM_Point
+ parameterToMeasure(): Set<MF_MeasureFunction>
+ positionAtParameter(Number*): MF_PositionExpression
+ prism(): GM_Curve

«type»
Geometry Types::MF_TemporalTrajectory

+ beginDomain: TM_Coordinate
+ endDomain: TM_Coordinate
+ graphTimeToAcceleration: GM_Curve [1..*]
+ graphTimeToCumulativeDistance: GM_Curve [1..*]
+ graphTimeToDistance: GM_Curve [1..*]
+ graphTimeToPoint: GM_Curve [1..*] = graphParameterT...
+ graphTimeToVelocity: GM_Curve [1..*]

+ acceleration(TM_Coordinate*): Vector
+ cumulativeDistanceAtTime(TM_Coordinate*): Distance
+ measureAtTime(TM_Coordinate*): Measure
+ nearestApproach(GM_Object*): Distance, TM_GeometricPrimitive[1..*]
+ offsetAtTime(TM_Coordinate*): Measure
+ pointAtTime(TM_GeometricPrimitive*): DirectPosition
+ positionAtTime(TM_Coordinate*): LR_PositionExpression
+ secondaryOffsetAtTime(TM_Coordinate*): Vector
+ subTrajectory(TM_Coordinate*, TM_Coordinate*): MF_TemporalTrajectory
+ timeAtCumulativeDistance(Distance*): TM_GeometricPrimitive
+ timeAtDistance(Distance*): TM_GeometricPrimitive
+ timeAtPoint(DirectPosition*): TM_GeometricPrimitive
+ timeToDistance(): GM_Curve[1..*]
+ timeToMeasure(): MF_MeasureFunction
+ velocity(TM_Coordinate*): Vector

«type»
Geometry Types::MF_TemporalGeometry

+ endTime(): TM_Coordinate
+ intersection(GM_Object*, TM_Period*): MF_TemporalGeometry
+ leafGeometry(TM_Coordinate*): GM_Object
+ nearestApproach(GM_Object*, TM_Period*): Distance, TM_GeometricPrimitive[1..*]
+ startTime(): TM_Coordinate
+ trajectory(DirectPosition*, TM_Coordinate*): MF_TemporalTrajectory

5.2.1. Basic Concepts

Volume and Boundary surface

ISO 19107 provides us with the GM_Object and GM_Boundary classes. GM_Object represents the space that is occupied by a geometry. GM_Boundary represents a surface which forms the boundary between inside and outside of the associated GM_Object. GM_Object and GM_Boundary are both specialized according to the number of dimensions they support.

Location and shape

The location of a Feature is a single GM_Point which represents its current position in the universe.

GM_Point is a geometry of zero dimensions, but is specified with values of all of the applicable coordinate axis.

The Shape of a Feature is the bounding geometry. Shape tells you nothing about where a Feature is but does specify the volume of the space it occupies.

Coordinate systems
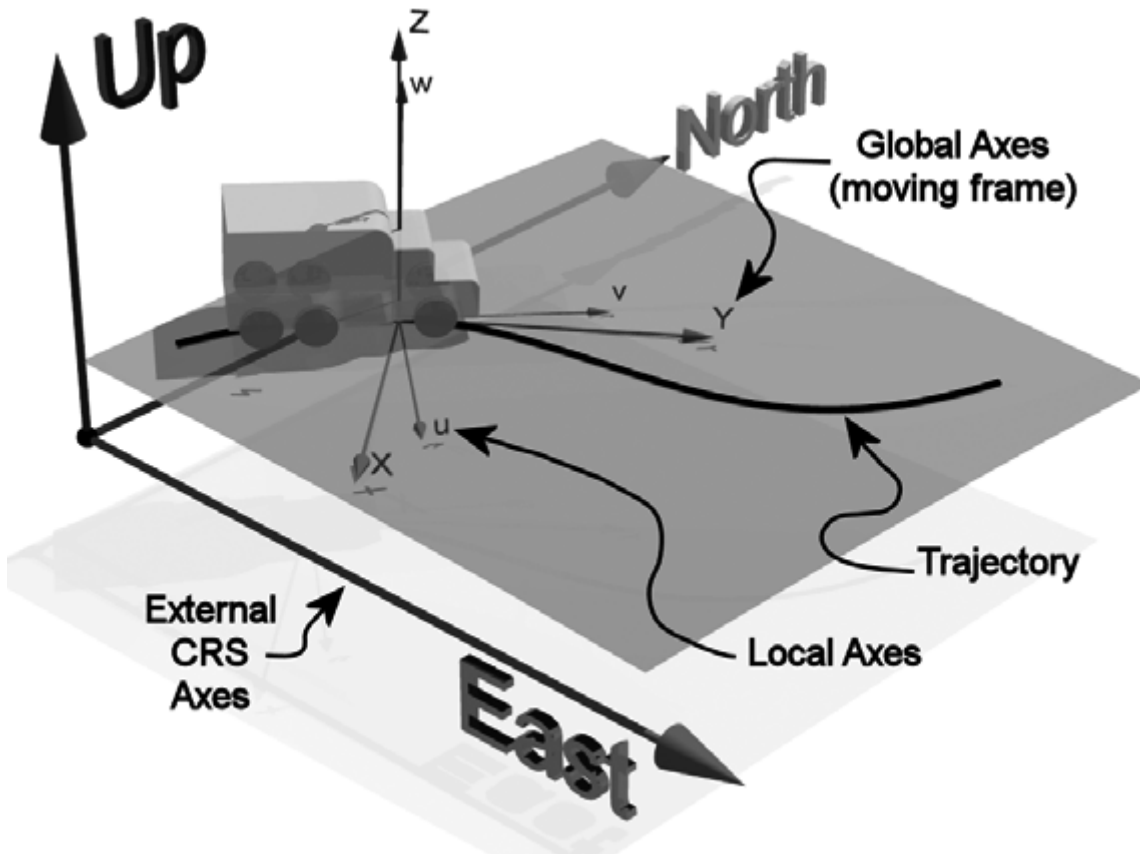
Moving Features deal with three spatial coordinate systems; External, Global, and Local.

The External coordinate system is the coordinate system within which the Moving Feature exists. Typically this is an Earth-centric geographic CRS.

The Global coordinate system is describes the area external to the Feature at a specific time. It is mostly used to identify the loctation and orientation of the Feature in regards to the path (tranjectory) in which it is traveling.

The local coordinate system is internal to the Feature. This is usually a cartisian coordinate system with the origin at a prominant point in the Feature such as the center of mass. The origin of the local coordinate system should also be the point where the location and orientation of the Feature is measured.

5.2.2. MF_OneParameterGeometry

We start our discussion with the class MF_OneParameterGeometry. MF_OneParameterGeometry is a subclass of GM_Object. So moving features have the 3D geometric properties of any other GM_Object. What is different is that this geometry can change as a function of a parameter. A one parameter set of geometries is defined as "a function f from an interval t Î [a, b] such that f(t) is a

geometry and for each point P Î f(a) there is a one parameter set of points (called the trajectory of P) P(t) : [a, b] ® P(t) such that P(t) Î f(t). A leaf of a one parameter set of geometries is the geometry f(t) at a particular value of the parameter".

A one parameter geometry instance includes a "leafgeometry()" operation. This operation takes the parameter (t) as input and returns the leaf P(t) for that parameter as a GM_Object.
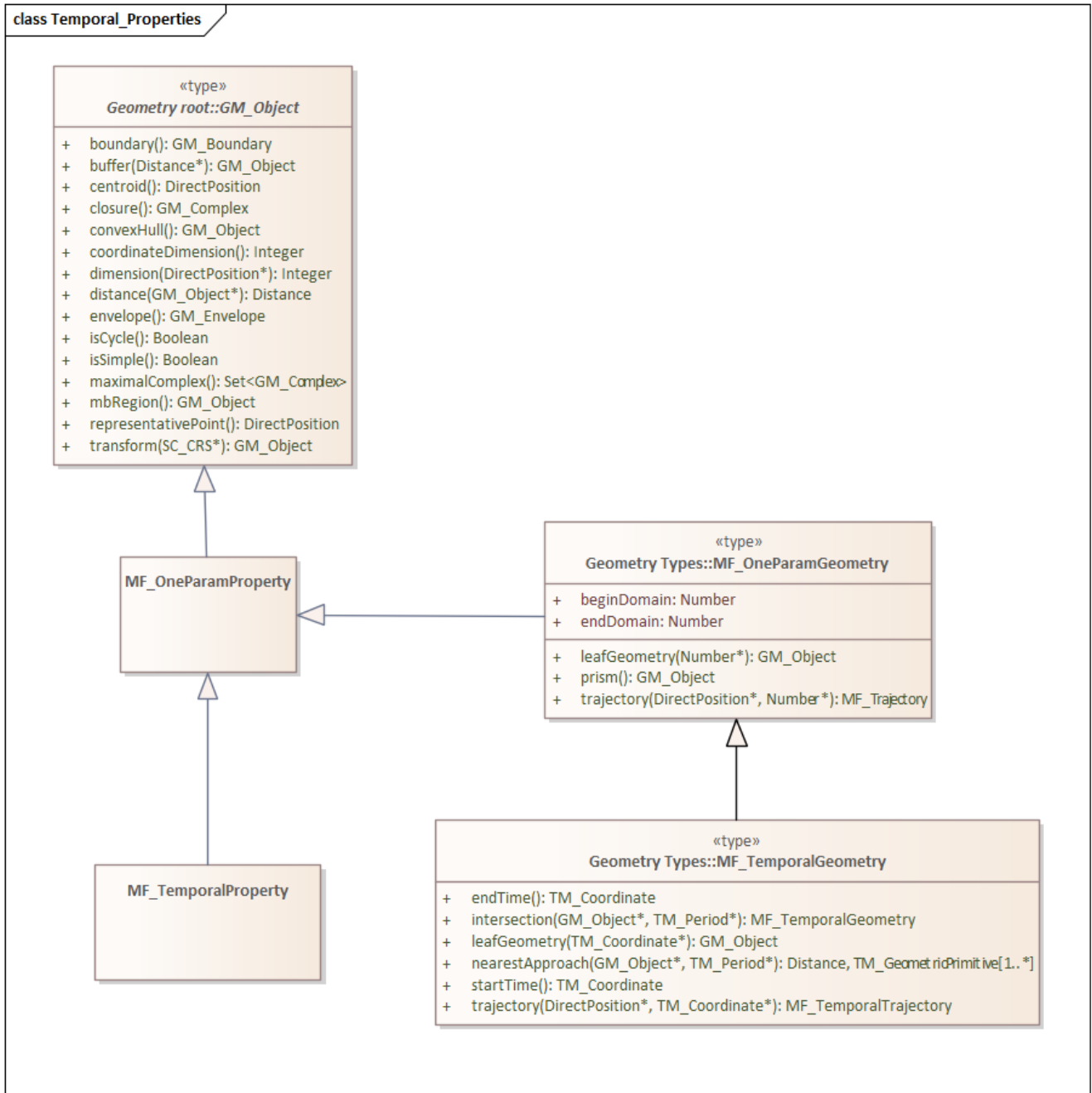
5.2.3. MF_TemporalGeometry

An MF_TemporalGeometry is a MF_OneParameterGeometry where the parameter is Time expressed as a TM_Coordinate. TM_Coordinate is specified in ISO 19108; it expresses time as a multiple of a single unit of measure such as year, day, or second. The "leafgeometry()" operation of an instance of MF_TemporalGeometry would take a TM_Coordinat in as input and return a GM_Object instance representing the geometry of the Feature at the specified point in time.

5.2.4. Temporal Properties

The JSON encoding of the OGC Moving Features standard introduces the concept of temporal properties.

"A TemporalProperties object is a JSON array of ParametricValues objects that groups a collection of dynamic non-spatial attributes and its parametric values with time."

Logically TemporalProperties should be a subclass of OneParamProperties. Since Geometry is a property, then MF_TemporalGeometry should be a subclass of TemporalProperties. Which gives us the following UML.

```
class Temporal_Properties

              «type»
         Geometry root::GM_Object

  +   boundary(): GM_Boundary
  +   buffer(Distance*): GM_Object
  +   centroid(): DirectPosition
  +   closure(): GM_Complex
  +   convexHull(): GM_Object
  +   coordinateDimension(): Integer
  +   dimension(DirectPosition*): Integer
  +   distance(GM_Object*): Distance
  +   envelope(): GM_Envelope
  +   isCycle(): Boolean
  +   isSimple(): Boolean
  +   maximalComplex(): Set<GM_Complex>
  +   mbRegion(): GM_Object
  +   representativePoint(): DirectPosition
  +   transform(SC_CRS*): GM_Object


  MF_OneParamProperty


                                «type»
                       Geometry Types::MF_OneParamGeometry

                     +   beginDomain: Number
                     +   endDomain: Number

                     +   leafGeometry(Number*): GM_Object
                     +   prism(): GM_Object
                     +   trajectory(DirectPosition*, Number*): MF_Trajectory


  MF_TemporalProperty

                                «type»
                       Geometry Types::MF_TemporalGeometry

                     +   endTime(): TM_Coordinate
                     +   intersection(GM_Object*, TM_Period*): MF_TemporalGeometry
                     +   leafGeometry(TM_Coordinate*): GM_Object
                     +   nearestApproach(GM_Object*, TM_Period*): Distance, TM_GeometricPrimitive[1..*]
                     +   startTime(): TM_Coordinate
                     +   trajectory(DirectPosition*, TM_Coordinate*): MF_TemporalTrajectory
```

Temporal properties are particularly useful for capturing state change. For example, the fuel load of an aircraft will change over time. The leafproperty() operation on a temporal fuel_load object would return the amount of fuel onboard at the specified time.

5.2.5. MF_TemporalTrajectory

A trajectory is a curve (GM_Curve) which represents the path that a Moving Feature takes as it moves. In other words, it represents the location of the Feature as a function of time.
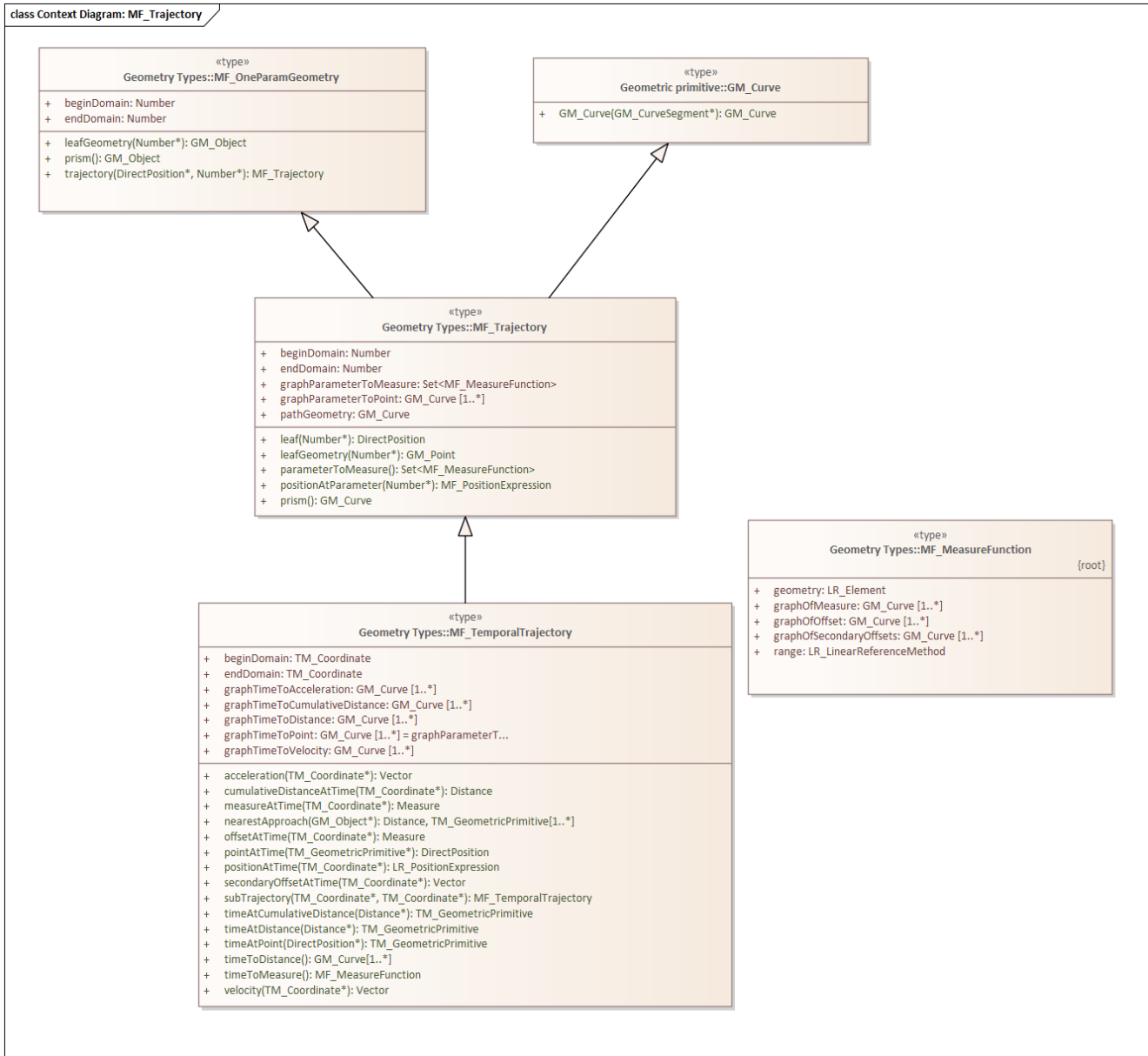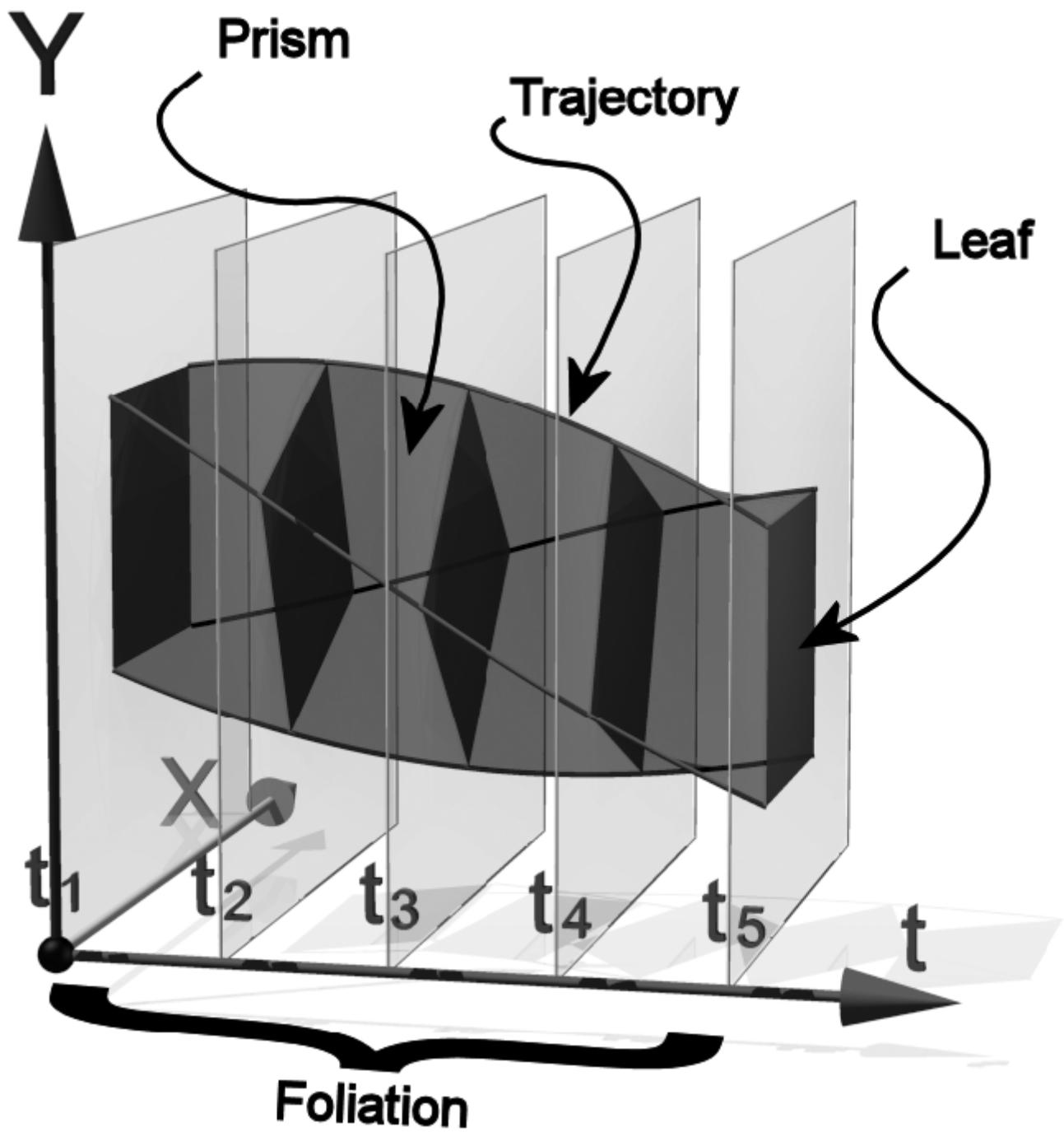
A Trajectory has two operations of particular interest; leaf() and leafgeometry(). The parameter for a Temporal Trajectory is always time (TM_Coordinate). One limitation of a trajectory is that it only represents location and orientation.
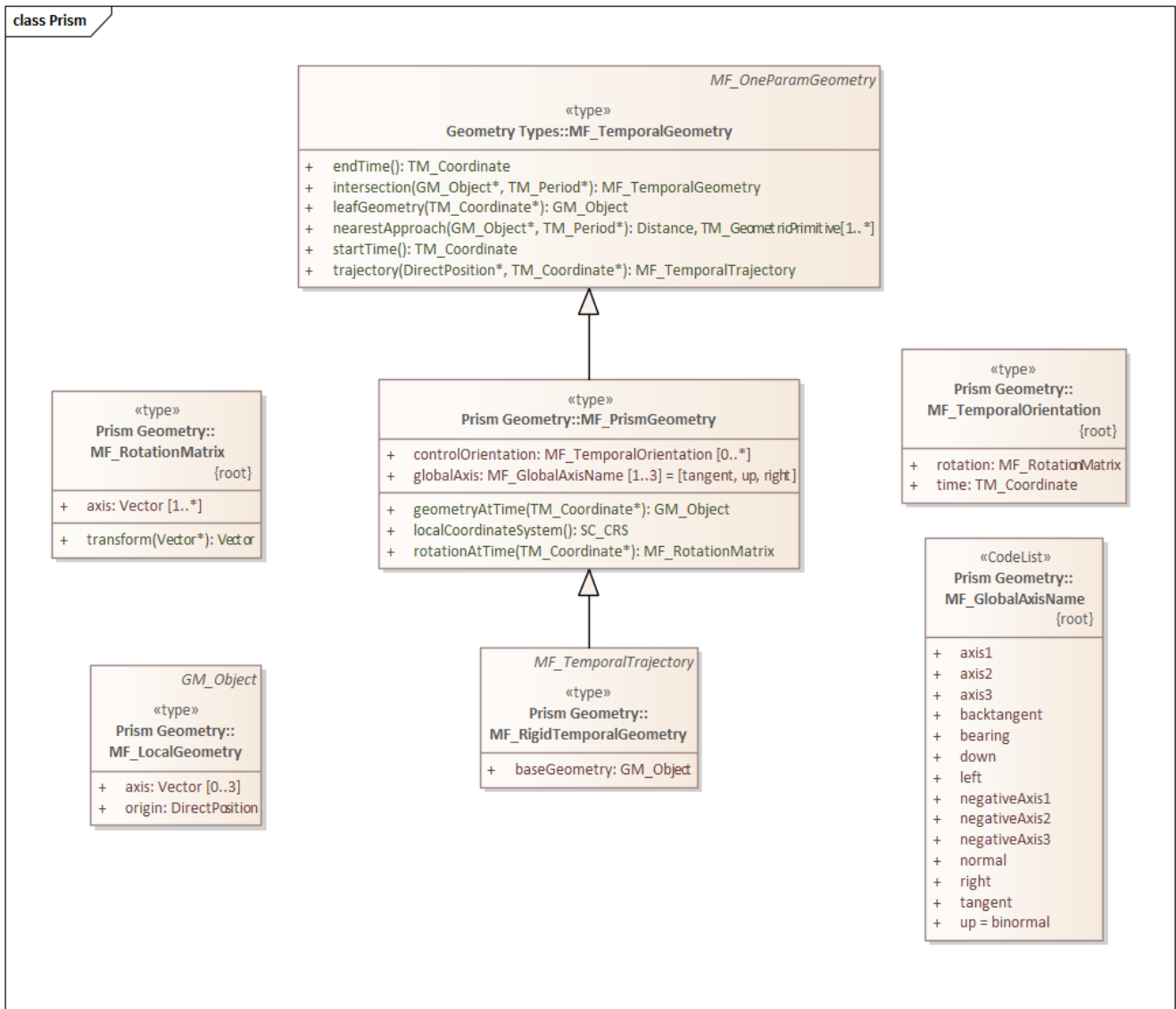
The leaf() operation returns the direct position that the trajectory passes at that value of the trajectory parameter. This is a point on the trajectory GM_Curve geometry. It typically serves as the origin of the Global CRS with the local tangent designating the X axis.

The LeafGeometry() operation returns the GM_Point that is at that position on the trajectory. This is the geometry of the Moving Feature expressed in the Local CRS. Since trajectories only convey location, only GM_Point geometries are supported.

5.2.6. MF_PrismGeometry

A Prism Geometry provides additional geometric information for a trajectory.

class Prism

MF_OneParamGeometry

«type»
Geometry Types::MF_TemporalGeometry

+ endTime(): TM_Coordinate
+ intersection(GM_Object*, TM_Period*): MF_TemporalGeometry
+ leafGeometry(TM_Coordinate*): GM_Object
+ nearestApproach(GM_Object*, TM_Period*): Distance, TM_GeometricPrimitive[1..*]
+ startTime(): TM_Coordinate
+ trajectory(DirectPosition*, TM_Coordinate*): MF_TemporalTrajectory

«type»
Prism Geometry::
MF_RotationMatrix
{root}

+ axis: Vector [1..*]

+ transform(Vector*): Vector

«type»
Prism Geometry::MF_PrismGeometry

+ controlOrientation: MF_TemporalOrientation [0..*]
+ globalAxis: MF_GlobalAxisName [1..3] = [tangent, up, right]

+ geometryAtTime(TM_Coordinate*): GM_Object
+ localCoordinateSystem(): SC_CRS
+ rotationAtTime(TM_Coordinate*): MF_RotationMatrix

«type»
Prism Geometry::
MF_TemporalOrientation
{root}

+ rotation: MF_RotationMatrix
+ time: TM_Coordinate

GM_Object

«type»
Prism Geometry::
MF_LocalGeometry

+ axis: Vector [0..3]
+ origin: DirectPosition

MF_TemporalTrajectory

«type»
Prism Geometry::
MF_RigidTemporalGeometry

+ baseGeometry: GM_Object

«CodeList»
Prism Geometry::
MF_GlobalAxisName
{root}

+ axis1
+ axis2
+ axis3
+ backtangent
+ bearing
+ down
+ left
+ negativeAxis1
+ negativeAxis2
+ negativeAxis3
+ normal
+ right
+ tangent
+ up = binormal

A trajectory provides the location of a Moving Feature as a function of time. Prism Geometry represents the full geometry (location, orientaion, and shape) of the Feature as a function of time.

Like a Temporal Trajectory, a Prism is a subclass of MF_TemporalGeometry. Prisms and Trajectories work together to provide this representation

The association role "originTrajectory" associates a Temporal Trajectory with a Prism. For any TM_Position, the associated Temporal Trajectory provides the location of the Moving Feature in the Global CRS. This point serves as the origin of the Local CRS.

The localCoordinateSystem() operation returns a SC_CRS for the design coordinate reference system in which the moving feature's shape is defined.

The rotationAtTime() operation accepts a time in the domain of the prism geometry and returns the rotation matrix that embeds the local geometry into geographic space at a given time (TM_Coordinate). The vectors of the rotation matrix allow the feature to be aligned and scaled as appropriate to the vectors of the global coordinate reference system.

This one association and two operations provide us with the location, orientation, axis definition, and units of measure needed to define identify the local CRS and to transform geometries between the Local and Global CRS.

Finally, the geometryAtTime() operation accepts a time in the domain of the prism geometry and returns the geometry of the moving feature, as it is at a given time in the global coordinate reference system. The return type is a GM_Object so this operation is not limited to points. It is fully capable of representing a 3D surface and volume.

5.2.7. Non-rigid Bodies

ISO 19141 only addresses rigid bodies. The shape returned by a geometryAtTime() operation will always be the same. However, it leaves open the opportunity to extend the Moving Feature model to support plastic (non-rigid) objects.

ISO 19141 postulates that trajectories can be used to represent non-rigid Features. This requires that each node on the geometry of the surface have its own trajectory. The shape of the Feature is defined by the bundle of trajectories for the surface points.

An alternate approach is to take advantage of temporal geometry to capture changes in the shape of a Feature as a function of time.

There are arguments for and against each approach. And an opportunity to do some valuable work.

5.3. Articulated Geometries

Given a suite of standards which allow you to define time-variant geometric elements, then the next step is to take a collection of those elements and assemble them into a complex object.

An articulated geometry is such an aggregation where each element has less than 6 degrees of freedom. Each element can move, but its movement is constrained by attachment to one or more additional elements.

The aggregate as a whole can also move, but that movement becomes more complex. typically we would model movement of the whole as a trajectory of the center of mass. However, the center of mass of an articulating Feature will change as the relative position of the elements change.

5.3.1. GeoPOSE

The Frame Transform is a representation of the transformation taking an Outer Frame coordinate system to an Inner Frame coordinate system. This abstraction is constrained in GeoPose v 1.0 to only allow transformations involving translation and rotation. The intention is to match the usual concept of a pose as a position and orientation. The formalism that expresses a GeoPose Frame Transform is a pair of Reference Frames, Outer and Inner, each defined by a Frame Specification.

A pose has a valid time (GeoPose_instant)

Discuss outer frame vs Inner Frame

EPSg 4979 is basis for all frames?

insert figure 8

Sequence

The sequence logical model defines a method for packaging of GeoPose data, where multipleGeoPoses in a sequence share the same Outer Frame and there is a time-dependent changing Inner Frame.

5.3.2. MISB Staging System

MISB 1906 Motion Imagery Metadata (MIMD): Staging System

"The Motion Imagery Metadata (MIMD) Model includes locations, orientations, and kinematics (velocity, acceleration, etc.) of platforms, gimbals, sensors, sensor elements, geospatial and other points. The locations and orientations are either absolute references to a well-known frame of reference (e.g., WGS84) or relative references to other locations and orientations. Each location and orientation pairing define a "stage" that has the potential to be the frame of reference for another location and orientation. Linking stages together forms the Staging System. The Staging System then defines an ability to describe, in metadata, the physical make-up and configuration of a system and the time varying physical relationships of the system and its sub-system components."

The MISB Staging System was designed to capture the instateneous configuration of an articulated sensor system.
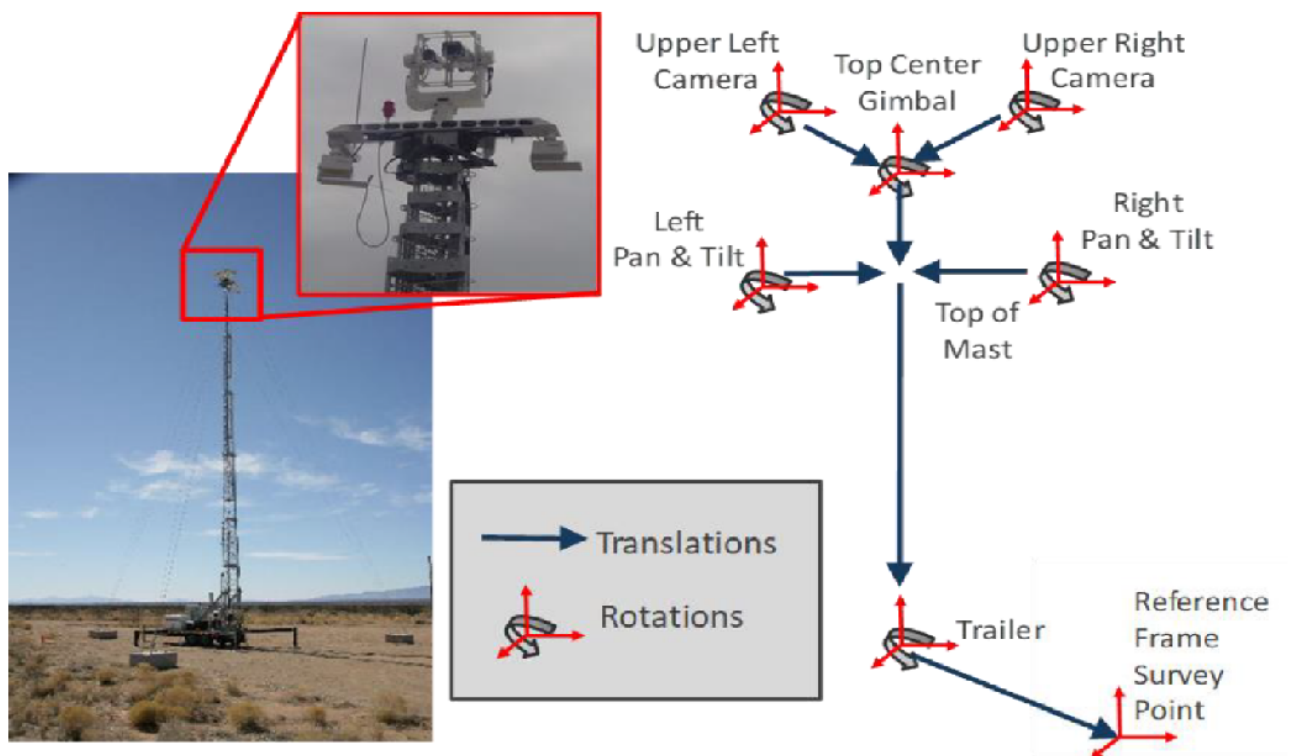
Example Motion Imagery System with Multiple Sensors and Gimbals (Photo credit White Sands Missile Range)

Stage: a single frame of reference located at a point. It defines the location, orientation, and kinemantics of a coordinate system located at that point. These properties can be defined in terms of absolute values, or as relative values measured from an "parent" reference system.

Constellation: A system of one or more stages where the parent-child relationships between the stages is sufficient to calculate the absolute values for every stage.

Root Stage: This is the starting point for a Constellation. This stage is expressed in absolute values.

Table 1. Stage System Example

Stage	Component	Parent	Values
1	Reference Frame Survey Point	0	Absolute Position/Orientation
2	Trailer	1	Relative Position / Orientation
3	Left Camera	2	Relative Position / Orientation
4	Right Camera	2	Relative Position / Orientation
5	Top Center Gimbal	2	Relative Position / Orientation
6	Left Upper Camera	5	Relative Position / Orientation
7	Right Upper Camera	5	Relative Position / Orientation

The Staging System's absolute positions use either WGS84 Ellipsoid angular coordinates (i.e., geodetic Latitude, Longitude, Height above Ellipsoid (HAE)), or WGS84 geocentric Earth-Centered Earth-Fixed (ECEF) Cartesian coordinates (i.e., X, Y, Z).

The Staging System's relative positions use Cartesian coordinates (i.e., x, y, z) measured in meters from the parent stage frame of reference.

The Staging System's orientations use Euler rotations, measured in radians, around three axes (X, Y, and Z) of a righthanded coordinate system. The Staging System's absolute orientation has the X, Y, and Z axes aligned with the East, North, and Up (ENU) respectively.

The Staging System standardizes rotations to use a specific order of rotations. The Staging System uses Tait-Bryan angles with a Primary Rotation Order of Z-Y-X (see Appendix C); thereby the first rotation is around the Z-axis, the second rotation is around the new Y-axis (after rotation around the Z-axis), the final rotation is around the new X-axis (after rotation around the Z and then Y axes).
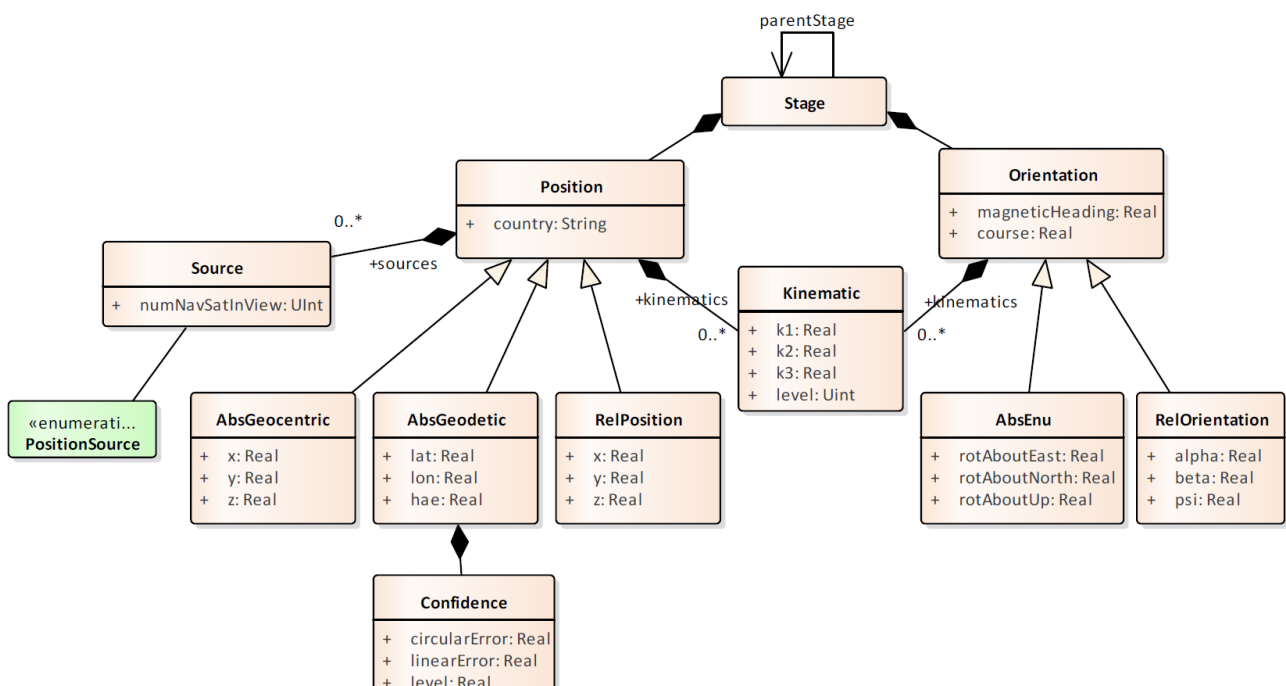
The Staging system does not explicity address temporality. However, this information is passed in the MISB KLV metadata stream. As such, each update is associated with the precision time stamp of that packet. In addition, an update can be associated with a Timer as described in section ----

5.3.3. Discussions

Quaternions vs. Euler rotations

Temporal representation

Standarize on a right-handed Cartisian coorinate system for all stages/Poses?

Common concepts:

1. An "anchor" node which ties the relative coordinates to an absolute CRS
2. Local coordinate systems associated with each other through standardized transformations

5.4. Mass Properties

Consider an articulated vehicle in free-flight. As the components change position, they change the center of mass, priciple axis of rotation, and a number of other mass properties. These changes affect the dynamic behavior of the vehicle. Most designers take great care to assure that problems cannot occur. But that doesn't mean we should ignore the issue.

Comments?

5.5. Temporal Reference Systems

Proposition: Time is in the eye of the beholder. So all measurements of time must be local.

Dynamic Features are not tied to an Earth-centered static existance. Yet the concepts of time used in the geospatial community are almost exclusivly based on Earth-centric astronomical phenomena. They also assume a rather coarse degree of granularity. For dynamic features we need to use local clocks with precision down to the nanosecond. We are less concerned with absolute time than with relative time. State B was achieved 37 nanoseconds after State A.

It's only when we begin aggregating these dynamic elements that we begin to worry about "absolute" time. Even then, we are more likely to convert from one local clock to another then to convert to an absolute time.

So if all time is local, we need a Temporal Reference System concept which captures the parameters needed to transform across TRS. A temporal equivalent to GeoPOSE.

Introduce the Timer concept from MISB.

Calendars and clocks

Local vs "Global" clocks

POSIX time is tied to a global datum, a calendar.

Local time is tied to a local counter.

Analogous to a Spatial Reference System vs. an Engineering CRS.

Is there an analog in the temporal world for the CRS transormation model?

5.6. Dynamic Features at Relativistic Velocities

At very high velocities, measurements between moving features sufer from relativistic effects. The proposition is to build on Minkowski SpaceTime to address these cases.

5.6.1. Lorentz Transformation

A transformation between two different coordinate frames that move at a constant velocity and are relative to each other.

ONly related to changes in inertial reference frames:

- Inertial frames - motion with a constant velocity
- Non-inertial frames - rotational motion with constant angular velocity, acceleration in curved paths.

In the reference frame "F" which is stationary, the coordinate defined are x, y, z, and t. In another reference frame F' which moves at a velocity v which is relative to F and the observer defines coordinate in the moving reverence frame as x', y', z', t'. In both the reference frames the coordinate axis are parallel and they remain mutually perpedicular. The relative motion is along the xx' axes. At t = t' = 0, the origins in both reference frames are the same (x,y,z) = (x',y',z') = (0,0,0).

The Lorentz factor g = 1 / (sqrt(1 - (v^2 / c^2)))

x' = g(x-vt) t' = g(t-(vx/c^2)) y' = y z' = z

This is not a complete coordinate transformation since F' has to be rotated and translated so as to be co-linear with F. However, it does add the impact that relative velocity (v) has on the measurements of x and t. In most cases this impact is neglegible (v^2 / c^2 approaches 0). However, when v is a significant percentage of c it should be applied.

5.6.2. Minkowski Space Time

It is basically a combination of 3-dimensional Euclidean Space and time into a 4-dimensional manifold, where the interval of spacetime that exists between any two events is not dependent on the inertial frame of reference.

Minkowski spacetime is a 4-dimensional coordinate system in which the axes are given by (x, y, z, ct)

Where ct is time (t) times the speed of light (c)

ds^2 = -c^{2dt}2 + dx^2 + dy^2, + dz^2 = the differential arc length in space time where:

1. dt = change in time

2. dx = change in x direction

3. dy = change in y direction

4. dz = change in z direction

Key point - while a Lorentz transformation deals with spatial measurements, Minkowski space includes time as part of that space-time. Thus ds is an arc length through space-time as opposed to a difference in x as in the Lorentz transform.

Question, since $c^{2dt}2$ is a negative term, does that inply that ct is an imaginary number orthogonal to x, y, and z (cti) such that i^2 = -1?

Yes for complex Minkowski space time. Here it is expressed as x^2 + y^2 + z^2 + (ict)^2 = const.

Complex Minkowski spacetime was replaced with real Minkowski space time where time is a real coordinate rather than an imaginary one.

Where v is velocity, and x, y, and z are Cartesian coordinates in 3-dimensional space, and c is the constant representing the universal speed limit, and t is time, the four-dimensional vector v = (ct, x, y, z) = (ct, **r**) is classified according to the sign of (c^2 t^2) − r^2. A vector is **timelike** if (c^2 t^2) > r^2, **spacelike** if (c^2 t^2) < r^2, and **null** or **lightlike** if (c^2 t^2) = r^2. This can be expressed in terms of the sign of η(v, v) as well, which depends on the signature. The classification of any vector will be the same in all frames of reference that are related by a Lorentz transformation (but not by a general Poincaré transformation because the origin may then be displaced) because of the invariance of the interval.

Annex A: Revision History

Date	Release	Editor	Primary clauses modified	Description
2016-04-28	0.1	G. Editor	all	initial version

Date	Release	Editor	Primary clauses modified	Description

Annex B: Bibliography

NOTE

Example Bibliography (Delete this note).

The TC has approved Springer LNCS as the official document citation type.

Springer LNCS is widely used in technical and computer science journals and other publications

- For citations in the text please use square brackets and consecutive numbers: [1], [2], [3]

– Actual References:

[n] Journal: Author Surname, A.: Title. Publication Title. Volume number, Issue number, Pages Used (Year Published)

[n] Web: Author Surname, A.: Title, http://Website-Url

[1] OGC: OGC Testbed 12 Annex B: Architecture. (2015).