

Interoperable Coordinate Transformation and Identification of Coordinate Systems

**Daniel Specht
Engineer Research and Development Center**

March 28, 2000

Interoperable Coordinate Transformation and Identification of Coordinate Systems

ABSTRACT

The OpenGIS Consortium (OGC), an industry group, has developed COM, CORBA and Java interface specifications allowing vendors to develop mutually interoperable geospatial software. OGC established the Coordinate Transformation Working Group (CT-WG) in April 1998. The working group developed a Unified Modeling Language object model of coordinate systems (CS) and coordinate transformations (CT) in coordination with ISO/TC211. In June 1999 OGC released this object model as part of a request for proposal. In February 2000 OGC accepted interface specifications (unpublished as of February 2000) drafted in response to this proposal.

This paper describes the object model and those specifications. The object model describes a CS as a collection of axes and at least one datum. A recent draft of the object model is available at <http://www.opengis.org/techno/request.htm> under "request 9."

The specification provides a common way of identifying CS and of accessing CT services that support accuracy calculation. When implemented, these specifications will ease data import; users of compliant applications will import data unaware of its coordinate system. If the application cannot import data in a given coordinate system, a compliant server will transform the coordinates to the native coordinate system.

INTRODUCTION

The OpenGIS Consortium (OGC) is a non-profit organization that promotes interoperable geoprocessing. The goal is that geospatial applications access data and services using standard interfaces. Clients using standard interfaces need not know much about the server. OGC defines OpenGIS as "transparent access to heterogeneous geodata and geoprocessing resources in a networked environment. The goal of the OpenGIS Project is to provide a comprehensive suite of open interface specifications that enable developers to write interoperating components that provide these capabilities." (<http://www.opengis.org>).

Representatives of software vendors, government agencies and academic institutions that belong to OGC write requirements for interface specifications. These requirements are published as a Request for Proposal (RFP). Member organizations then submit proposals for these specifications. After OGC accepts a proposal as an Implementation Specification OGC submits it to the International Standards Organization Technical Committee 211, ISO/TC211 (<http://www.statkart.no/isotc211/>) for acceptance as an ISO standard.

Interface standards are widely used outside of the geospatial community. For example in the Microsoft environment a user can cut out a piece of a spreadsheet and drop it into any conformant application (e.g., MS Word). The application will have access to spreadsheet services and data stored on the spreadsheet. To the user it appears as if the spreadsheet is running on the application. What we cannot do yet is to tell our spreadsheet that column H is Transverse Mercator coordinates and please change them to Lambert coordinates using a CT service on the network.

However OGC has developed several sets of geospatial interfaces. For example users with two applications compliant with the Simple Features specification for COM could (in theory) cut and paste a geospatial feature from one application to the other running on the same network.

THE ABSTRACT MODEL OF COORDINATE TRANSFORMATION

The US Army struggled with poor CT software for years. Interoperable CT software would help solve this problem. With this in mind I chartered a CT Working Group (CT-WG) at OGC. The group developed the Abstract Model, an object model of CT and CS using Unified Modeling Language (UML), a notational language used for object oriented analysis and design. UML is widely used and has been standardized by the Object Modeling Group (<http://www.omg.org>). Figure 1 shows what a UML object class looks like.

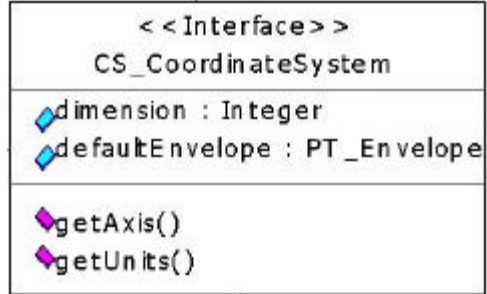


FIGURE 1. The class has three parts. The name of the class is in the top part, the middle part lists attributes (or properties) of the class and the lower part lists methods, i.e. implementable functions. In this case the methods get metadata, the axis and units.

ACHIEVING CONSENSUS

The CT-WG struggled to reach a consensus on what the model should look like while coordinating with our counterparts at ISO/TC211 to maintain conformance between the ISO/TC211 model and our own. This was not always easy.

Consider a point stored as latitude, longitude and elevation above sea level. Latitude and longitude are referenced to a geodetic datum while elevation is referenced to a vertical datum.

ISO/TC211 calls this a compound coordinate reference system, composed of two coordinate reference systems, each with its own datum, shown in Figure 2. (The diamonds on a stick indicate strong aggregation; the datum is part of the Coordinate Reference System which is part of the Compound Coordinate Reference System.)

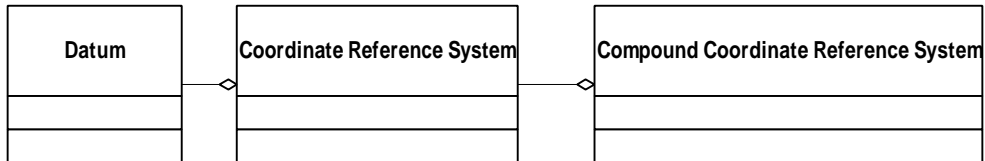


FIGURE 2. An ISO view

Some OGC members felt this was a single coordinate reference system with two datums and that the model should not include a compound coordinate reference system (Figure 3). There were other proposals as well.

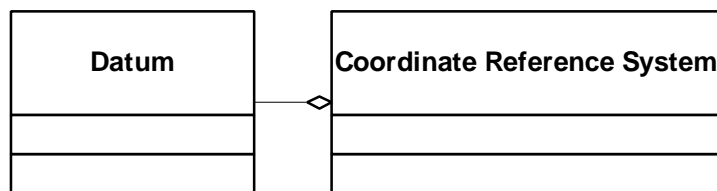


FIGURE 3: An early OGC view

The CT-WG could not come to a consensus and created a UML model that left the issue open. This UML model is part of the OpenGIS Abstract Specification, <http://www.opengis.org/public/abstract/99-102r1.pdf> which was included in the RFP.

OGC asked members to propose an Implementation Specification that solves this problem. The proposed solution is shown in figures 4 and 5. The revised UML model will be published shortly as part of the implementation specification.

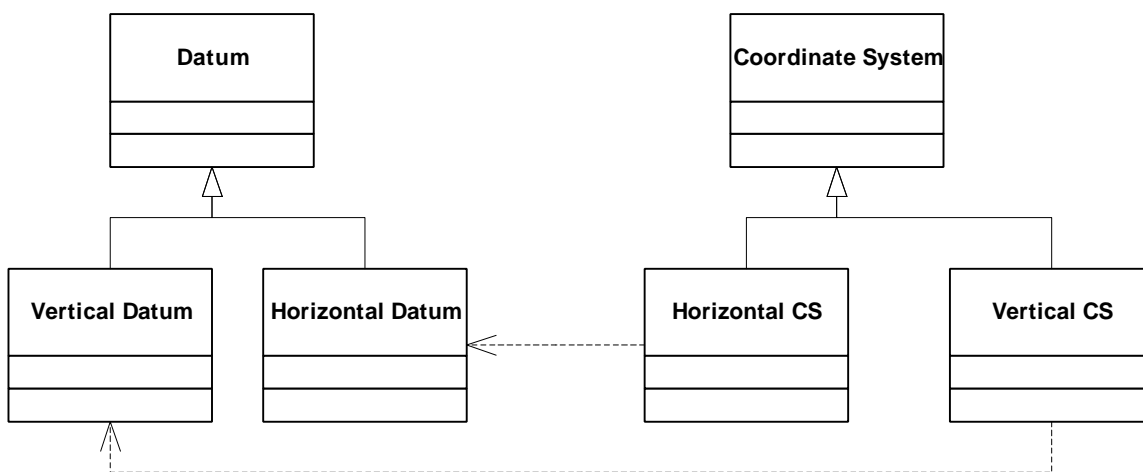


FIGURE 4. A representation of the final model. The object at the pointy end of the triangle is the superclass and the object at the other end is the subclass. The dotted line with the arrows indicates dependency; if the datum goes away so does the CS. These relationships are not explicit in the actual UML model, part of which is shown in Figure 5.

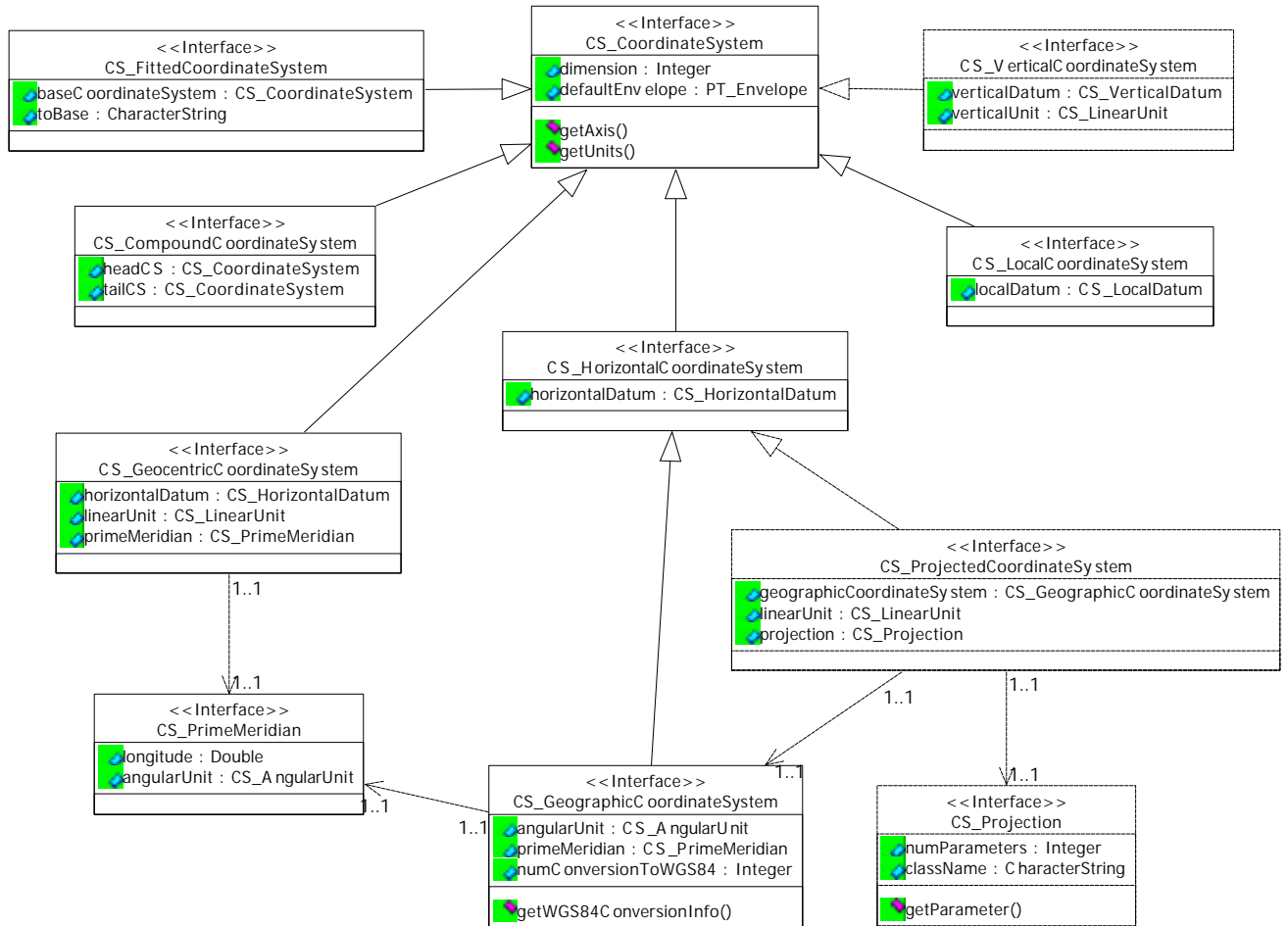


FIGURE 5. Here there is no association between the datum superclass and the CS superclass. Instead there are relationships between subclasses, e.g., a Vertical CS and a Vertical Datum. This model uses the term Coordinate System rather than Coordinate Reference System

IMPLEMENTATION MODEL

The object model for CS includes a number of CS classes and a number of datum classes.

A **Compound CS** includes a horizontal and a vertical CS and references two datums, a horizontal (geodetic) and a vertical datum. However where the Compound CS is 3-D ellipsoidal both the horizontal and vertical components may reference the same datum.

A **Horizontal CS** is a 2-D CS that may be **Projected** (a cartographic projection) or **Geographic** (latitude and longitude). Horizontal CS use horizontal (geodetic) datums.

A **Vertical CS** is a vertical 1-D CS (e.g., vertical with respect to a plumb line or to an ellipsoid). This references a vertical datum such as a sea level datum or a horizontal (geodetic) datum.

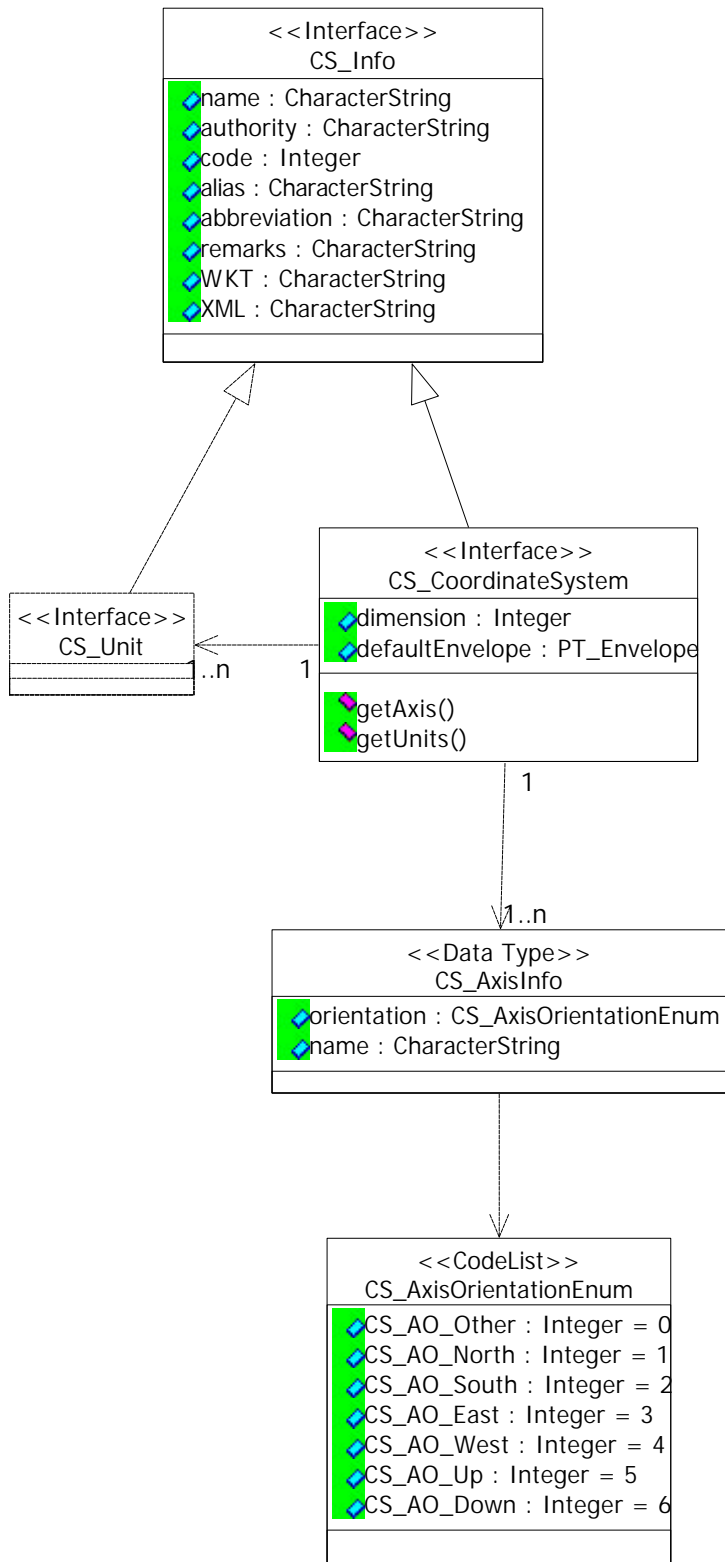


FIGURE 6. This specification supports users who create a coordinate system in any number of dimensions.

A **Local CS** references a local datum. A Local CS cannot be transformed to another CS. Once the Local CS is georeferenced it is no longer a Local CS. Note that a local datum is not necessarily 2-D Cartesian.

Geocentric CS are 3-D Cartesian geocentric CS that reference a horizontal datum.

Fitted CS sit “inside another CS. The fitted CS can be rotated and shifted, or use any other math transform [a mathematical function] to inject itself into the base CS.” (unpublished draft specification). A Fitted CS references the datum of the base CS.

This specification will also support creation of coordinate systems of any dimension, because each of these CS and datum have the metadata shown in figure 6, representing ellipsoid, geoid, prime meridian, axis, parameters (both projection and transformation parameters) and unit (i.e. linear or angular units). The specification also includes enumerated lists for things like datum type (e.g., Altitude Barometric and Orthometric) and axis orientation. Attributes in CS_Info are from ISO/TC211 Spatial Referencing by Coordinates.

IDENTIFICATION

Interoperability requires a universal way to identify a CS, even if the CS is unique to a small project. A client cannot import data without knowing the CS. However no one has volunteered to keep the list of all CS for the entire geospatial community. The best anyone can do is to keep a list of lists.

Our CS identifier consists of an authority (one of the lists within the list) and a code (an integer). The authority is whoever assigned the integer to the CS. For example the European Petroleum Survey Group (EPSG, <http://www.petroconsultants.com/products/epsg21.html>) assigned 26778 to the Kansas South state plane coordinate system, so our CS identifier is EPSG:26778. This system is distributed, extensible and supports legacy name lists (if they are translated into integers).

INTERFACES

There are interfaces for CT (not discussed in this paper), to create a CS and to access CS metadata (i.e., attributes or properties). The interfaces support accuracy of a transformation and of a coordinate, but not precision of a CS.

The specification includes three profiles (i.e., versions), Microsoft Interface Definition Language (MIDL) specifications for COM interfaces, Interface Definition Language (IDL) specifications for CORBA interfaces and Java source specifications for Java interfaces. All three profiles have analogous functionality.

CONCLUSION

The CT specification will provide a common way of identifying CS and of accessing CT services that support accuracy calculation. When implemented, these specifications will ease data import. Users of compliant applications will import data without having to be aware of the CS of the imported data because the client and server will use the same identifier for a given CS. If the application cannot transform imported data to the native CS, a compliant server will transform the coordinates to the native CS. Compliant software will therefore import data more reliably and be able to deal more effectively with distributed data.

Compliant software is already emerging. A GIS vendor (Cadcorp Ltd, UK) has already fully

implemented the COM profile of the CT specification. The specification supports the software's ability to transform an image or a feature (i.e. a feature geometry). Figure 7 shows how this prototype Cadcorp product has transformed an image.

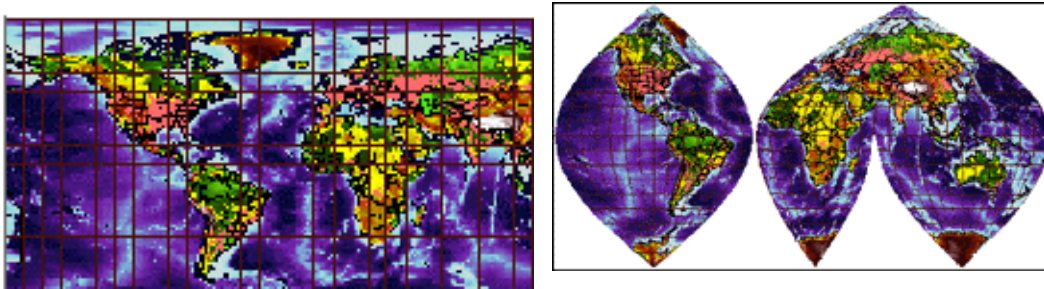


FIGURE 7

The image on the left in figure 7 was downloaded using the Open GIS Web Map Server Interface Specification (http://www.opengis.org/info/press/wm_overlays_pr.htm). Within months web browsers will overlay geospatial data views from various and diverse web servers that use this specification. Overlay will be possible because of interoperable coordinate transformation software.

DOD plans to modify its library of CT functions and its CT application (GeoTrans, available from iatbsw@tec.army.mil) so that it conforms to the OGC CT (Java) interface specifications. GeoTrans could serve CT on the web to any compliant application. Compliant applications would use the CT service invisibly to the user.

OGC maintains the CT specifications, improving them based on feedback from software developers and developing additional CT interface specifications.

REFERENCES

Geographic information - Spatial referencing by coordinates, (Committee Draft) project no. 19111 (15046-11), N814, 1999, ISO/TC211 WG3

OGC Request 9: Core Task Force, Coordinate Transformation Working Group, A Request for Proposals: OpenGIS Coordinate Transformation Services, 1999, OpenGIS Consortium

OpenGIS Implementation Specification: Coordinate Transformation Services, OpenGIS Project Document, 2000, Cadcorp Ltd. (unpublished)

UML Distilled, Applying the Standard Object Modeling Language, Fowler, M. and Scott, K., 1997, Addison Wesley