BAE SYSTEMS

BAE Systems
10920 Technology Pl.
San Diego CA 92127

Document No:	2100501
CAGE Code:	12436
Date:	15 April 2022

Mensuration Services Program (MSP)

Programmer's Guide for

MSP Geographic Translator (GEOTRANS) Version 3.9

Program Title:	Radiance

Contract No:	HM0476-17-D-0015
CDRL:	A027
DID:	Contractor Format

Prepared For:	National Geospatial-Intelligence Agency
	7500 GEOINT Drive
	Springfield, VA 22150

Change Log

Version	Date	Description of Change
01	3 December 2009	Original issue.
02	22 September 2010	Updated for release MSP GeoTrans 3.1 reflecting a change to the supported platforms, the inclusion of static libraries in the release, the directory of files organization, support for the EGM84 30 minute interval geoid separation grid, the addition of sample code to the release, and updated instructions for building on Windows.
03	06 October 2011	Updated for release MSP GeoTrans 3.2 reflecting the addition of EGM 2008.
04	23 October 2012	Updated for release MSP GeoTrans 3.3.
05	14 November 2013	Updated for release MSP GeoTrans 3.4.
06	23 October 2014	Updated for release MSP GeoTrans 3.5.
07	10 September 2015	Updated for release MSP GeoTrans 3.6.
08	18 February 2016	Updated for release MSP GeoTrans 3.7.
09	10 May 2019	Updated for release MSP GeoTrans 3.8.
10	15 April 2022	Updated for release MSP GeoTrans 3.9.

Document No. 2100501
15 April 2022

Table of Contents

Document No. 2100501
15 April 2022

Document No. 2100501
15 April 2022

List of Figures

Document No. 2100501
15 April 2022

1. SCOPE

1.1 IDENTIFICATION

This document is the Programmer's Guide for Version 3.9 of the MSP Geographic Translator (GEOTRANS).

1.2 SYSTEM OVERVIEW

GEOTRANS (Geographic Translator) is an application that allows you to convert geographic coordinates among a wide variety of coordinate systems, map projections, grids, European Petroleum Survey Group (EPSG) codes, and datums. GEOTRANS runs in Microsoft Windows, and LINUX environments.

The user interface of GEOTRANS consists primarily of a single window. To convert coordinates, select the EPSG code or the coordinate reference frame and datum in which your coordinates are defined, enter any associated parameters, and enter the coordinates in the upper half of the window. Then select the EPSG code or the coordinate reference frame and datum to which you want the coordinates to be converted, and enter any associated parameters, in the lower half of the window. Click on the Convert Upper-to-Lower button, and the resulting coordinates will be displayed in the lower half of the window. You can convert additional coordinate sets from the same source by just entering the new coordinates and clicking on the Convert Upper-to-Lower button. You can change any of the coordinate reference frame, datum, or parameter selections at any time. Also, you can reverse the roles of input and output by using the Convert Lower-to-Upper button. Currently, forty-one different types of coordinate systems, map projections, grids, and coding schemes are supported, as well as more than two hundred different horizontal datums and one hundred thirty EPSG codes.

GEOTRANS can also be used to efficiently convert large numbers of coordinates contained in text files. The file format is very simple. A multi-line file header defines the coordinate reference frame and datum of the coordinates contained in the file, including any associated parameter values. Following the header, each line contains a single set of coordinates, separated by commas followed by at least one space. Using the GEOTRANS file processing interface, you can select an existing file of coordinates to be converted. You can then define the coordinate reference frame and datum to which you want to convert the coordinates, along with any associated parameter values. Finally, you can specify the name and location of the output file that is to be created.

GEOTRANS then converts all of the coordinates in the input file and creates the output file as a single operation.

1.3 DOCUMENT OVERVIEW

The remainder of this document is organized as follows:

- Section 2 identifies the other documents that are referenced in this Programmer's Guide.

- Section 3 describes the programming environment in which the GEOTRANS application was developed, and summarizes the structure of the GEOTRANS application software.

- Section 4 presents the as built design of the GEOTRANS application software, briefly describing each of the components, the build procedures, and modification procedures.

- Section 5 contains additional notes.

Document No. 2100501
15 April 2022

2. REFERENCES

The following documents describe coordinate systems, map projections, coordinate conversion, and datum transformation:

1. NGA.STND.0037_2.0.0_GRIDS, **Universal Grids and Grid References Systems, v2.0.0,** 28 February 2014.

2. NGA.SIG.0012_2.0.0_UTMUPS, **The Universal Grids and the Transverse Mercator and Polar Stereographic Map Projections, v2.0.0**, 25 March 2014.

3. NGA.STND.0036_1.0.0_WGS84, **World Geodetic System 1984: Its Definition and Relationship with Local Geodetic Systems, V1.0.0**, 08 July 2014.

4. STANAG 2211, **Geodetic Datums, ellipsoids, grids and grid references**, Sixth Edition, 2001.

5. TEC-SR-7, **Handbook for Transformation of Datums, Projections, Grids and Common Coordinate Systems**, 1996.

6. NGA, **Department of Defense Glossary of Mapping, Charting, and Geodetic Terms**, Fourth Edition, 1981.

7. Rapp, Richard H., **Geometric Geodesy - Part I; Department of Geodetic Science and Surveying**, The Ohio State University, Columbus, Ohio. 1984.

8. Rapp, Richard H., **Geometric Geodesy - Part II; Department of Geodetic Science and Surveying**, The Ohio State University, Columbus, Ohio. 1987.

9. Snyder, J. P., **Geological Survey Professional Paper 1395 Map Projections - A Working Manual**, 1987.

10. Federal Geographic Data Committee (FGDC), FGDC-STD-011-2001, **United States National Grid**, December 2001

11. http://earth-info.nga.mil/GandG/coordsys/grids/gars.html, **Global Area Reference System (GARS)**

12. **ISO/IEC 18026:2009 Information Technology - Spatial Reference Model (SRM)**

Document No. 2100501
15 April 2022

3. PROGRAMMING ENVIRONMENT

This section describes the programming environment in which the GEOTRANS application was developed, and summarizes the structure of the GEOTRANS application software. It also summarizes the hardware requirements of the GEOTRANS application software.

3.1 SOFTWARE

3.1.1 OPERATING SYSTEM(S)

The GEOTRANS software was developed and tested using the Microsoft Windows 10 operating systems, and Red Hat LINUX. It should work on all later versions of these operating systems. The software was designed to reduce as much as possible any operating system dependencies.

3.1.2 COMPILATION AND LINKING

The GEOTRANS Coordinate Conversion Service (CCS) was developed in C++. The Windows version was developed and tested on the Windows 10 32-bit platform using the Microsoft Visual Studio 2015 compiler. The Windows version was also developed and tested on the Windows 10 64-bit platform using the Microsoft Visual Studio 2015 compiler. The Linux version was developed and tested on the Red Hat Enterprise Linux (RHEL) v7 64-bit platform using the GNU C++ compiler (GCC) v4.8.5.

The GEOTRANS application Graphical User Interface (GUI) was developed in Java. The Java GUI was developed using the NetBeans Integrated Development Environment (IDE), available at www.netbeans.org.

3.1.3 DIRECTORY STRUCTURE

The GEOTRANS software distribution includes all of the directories and files needed to build and execute all versions (LINUX and Windows) of the GEOTRANS software. The GEOTRANS software distribution consists of three top-level components:

1. The GEOTRANS Application software component is contained in a directory named `GEOTRANS3`.

2. The GEOTRANS Coordinate Conversion Service software component is contained in a directory named `CCS`.

3. The GEOTRANS data files needed for initializing the GEOTRANS application are contained in a directory named `data`.

4. GEOTRANS example batch input files are contained in a directory named **Examples**.

5. GEOTRANS test code, inputs/outputs and procedures are contained in a directory named **SpreadsheetTester**.

3.1.3.1 GEOTRANS Application Component

Within the `GETOTRANS3` directory, the GEOTRANS Application software component is organized into subdirectories as follows:

- `docs` – contains documentation for the GEOTRANS application software in Adobe Portable Document Format (PDF).

- `help` – contains the on-line help source files for the GEOTRANS application, including files in HyperText Markup Language (HTML) and graphics files in Graphic Interchange Format (GIF) format.

- `java_gui` – contains the source code for the GEOTRANS Application graphical user interface GUI.
- `linux_64` – contains the LINUX GEOTRANS jar file, the JNI CCS shared object and archive libraries, makefile, and the runGeotrans.sh startup script.

- `win` – contains the Windows GEOTRANS jar file, the 32-bit JNI CCS Dynamic Link Library (DLL) and Static Linked Library (LIB), Visual C++ solution and project files, and the runGeotrans.bat startup script.

- `win_64` – contains the Windows GEOTRANS jar file, the 64-bit JNI CCS Dynamic Link Library (DLL) and Static Linked Library (LIB), Visual C++ solution and project files, and the runGeotrans.bat startup script.

3.1.3.2 GEOTRANS Coordinate Conversion Service Component

The GEOTRANS Coordinate Conversion Service resides within the `CCS` directory. All input and output angular parameters are in units of radians. The CCS supports the following types of coordinate conversions:

- Conversions between geodetic, geocentric, spherical, and local Cartesian coordinate systems.

- Conversions between geodetic coordinates (latitude and longitude in radians) and various types of map projection coordinates (easting and northing in linear units).

- Conversions between geodetic coordinates and Military Grid Reference System (MGRS), United States National Grid (USNG), GARS, or World Geographic Reference System (GEOREF) grid coordinates.

- Transformations of geodetic or geocentric coordinates between different global or local horizontal datums.

- Transformations between ellipsoid heights and geoid (or local MSL) heights.

- Conversions that combine two or more of the above operations.

Within the **CCS** directory, the GEOTRANS Coordinate Conversion Service software component is organized into subdirectories as follows:

- **docs** – contains the CCS Application Programming Interface (API) documentation in the form of a set of hyperlinked HTML files created using the doxygen documentation generation tool.

- **src** – contains the source files for the CCS component.

- **linux_64** – contains the LINUX CCS shared object and archive libraries, and the makefile needed to build the CCS shared object libraries for LINUX.

- **win** – contains the 32-bit Windows CCS DLL and LIB libraries and the Visual C++ solution and project files needed to build the CCS Windows DLL.

- **win_64** – contains the 64-bit Windows CCS DLL and LIB libraries and the Visual C++ solution and project files needed to build the CCS Windows DLL.

3.1.3.3 GEOTRANS Data Component

Within the **data** directory are the datum and ellipsoid parameter files, EPSG codes csv file and the geoid separation grid files, required to properly initialize the GEOTRANS software.

3.2 HARDWARE REQUIREMENTS

The hardware requirements of the GEOTRANS software are minimal. It is designed to run on any 32- or 64-bit processor with 512MB or more of memory, and 300MB or more of available disk space.

3.3 PACKAGING REQUIREMENTS

The GEOTRANS software can be downloaded from NGA's Office of Geomatics webpage on the world wide web (WWW): https://earth-info.nga.mil/.

Additionally, the software can be downloaded from the SIPRNet or JWICS websites, or ordered on CD-ROM or DVD media via the MSP Help line.

SIPRNet link: http://www.geoint.nga.smil.mil/products/gandg/update/index.php

JWICS link: https://www.geoint.nga.ic.gov/products/gandg/update/index.php

MSP Help via e-mail at MSP_Help@nga.mil.

4. SOFTWARE SUPPORT INFORMATION

This section describes the as built design of the GEOTRANS software.

This section describes each of the components that make up the GEOTRANS software. The GEOTRANS software consists of three components, as follows:

- GEOTRANS Coordinate Conversion Service software is located in the **CCS** directory. Sample code, makefiles and startup scripts are located in the **CCS/samplecode** directory.

- GEOTRANS Application software is located in the **GEOTRANS3** directory.

- GEOTRANS data files are located in the **data** directory.

Each of these components is described below.

4.1 GEOTRANS COORDINATE CONVERSION SERVICE SOFTWARE

This subsection describes the structure and organization of the GEOTRANS Coordinate Conversion Service software, which is located in the **CCS/src** directory. The **CCS/src** directory is organized into subdirectories, as follows:

- **CoordinateConversion** – contains the source code for the CoordinateConversionService class.

- **dtcc –** contains the DatumLibrary, EllipsoidLibrary, and GeoidLibrary classes. These classes and the CoordinateConversionService class form the primary external interface of the GEOTRANS Coordinate Conversion Service. They are shown in Universal Markup Language (UML) form in Figure 4-1.
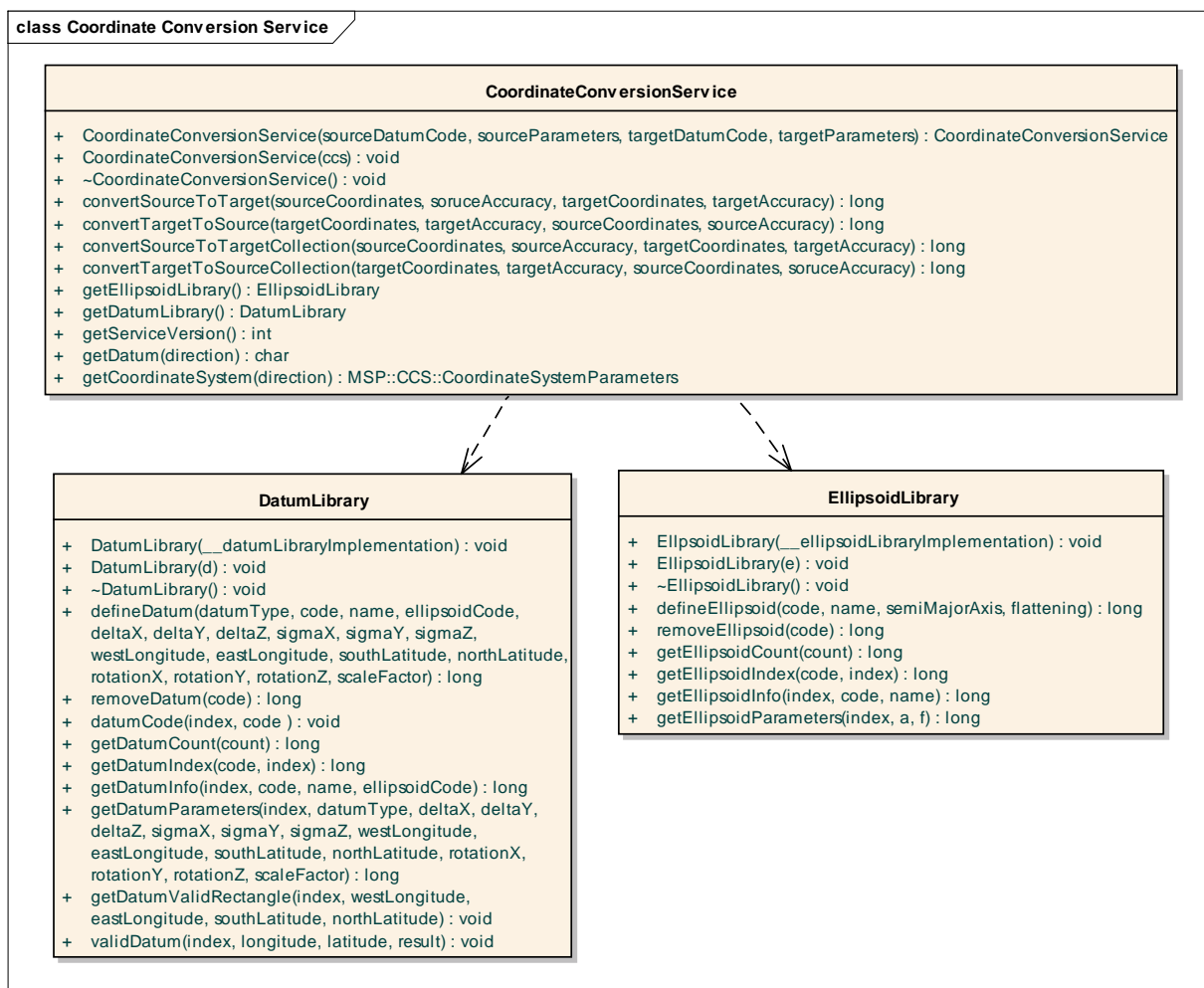
class Coordinate Conversion Service

CoordinateConversionService

+ CoordinateConversionService(sourceDatumCode, sourceParameters, targetDatumCode, targetParameters) : CoordinateConversionService
+ CoordinateConversionService(ccs) : void
+ ~CoordinateConversionService() : void
+ convertSourceToTarget(sourceCoordinates, soruceAccuracy, targetCoordinates, targetAccuracy) : long
+ convertTargetToSource(targetCoordinates, targetAccuracy, sourceCoordinates, sourceAccuracy) : long
+ convertSourceToTargetCollection(sourceCoordinates, sourceAccuracy, targetCoordinates, targetAccuracy) : long
+ convertTargetToSourceCollection(targetCoordinates, targetAccuracy, sourceCoordinates, soruceAccuracy) : long
+ getEllipsoidLibrary() : EllipsoidLibrary
+ getDatumLibrary() : DatumLibrary
+ getServiceVersion() : int
+ getDatum(direction) : char
+ getCoordinateSystem(direction) : MSP::CCS::CoordinateSystemParameters

DatumLibrary

+ DatumLibrary(__datumLibraryImplementation) : void
+ DatumLibrary(d) : void
+ ~DatumLibrary() : void
+ defineDatum(datumType, code, name, ellipsoidCode, deltaX, deltaY, deltaZ, sigmaX, sigmaY, sigmaZ, westLongitude, eastLongitude, southLatitude, northLatitude, rotationX, rotationY, rotationZ, scaleFactor) : long
+ removeDatum(code) : long
+ datumCode(index, code) : void
+ getDatumCount(count) : long
+ getDatumIndex(code, index) : long
+ getDatumInfo(index, code, name, ellipsoidCode) : long
+ getDatumParameters(index, datumType, deltaX, deltaY, deltaZ, sigmaX, sigmaY, sigmaZ, westLongitude, eastLongitude, southLatitude, northLatitude, rotationX, rotationY, rotationZ, scaleFactor) : long
+ getDatumValidRectangle(index, westLongitude, eastLongitude, southLatitude, northLatitude) : void
+ validDatum(index, longitude, latitude, result) : void

EllipsoidLibrary

+ EllpsoidLibrary(__ellipsoidLibraryImplementation) : void
+ EllipsoidLibrary(e) : void
+ ~EllipsoidLibrary() : void
+ defineEllipsoid(code, name, semiMajorAxis, flattening) : long
+ removeEllipsoid(code) : long
+ getEllipsoidCount(count) : long
+ getEllipsoidIndex(code, index) : long
+ getEllipsoidInfo(index, code, name) : long
+ getEllipsoidParameters(index, a, f) : long

Figure 4-1. CoordinateConversionService, DatumLibrary, and EllipsoidLibrary

- **dtcc/CoordinateSystemParameters** – contains the source code for the abstract CoordinateSystemParameters class and all of its concrete subclasses. These classes are shown in UML form in Figure 4-2.
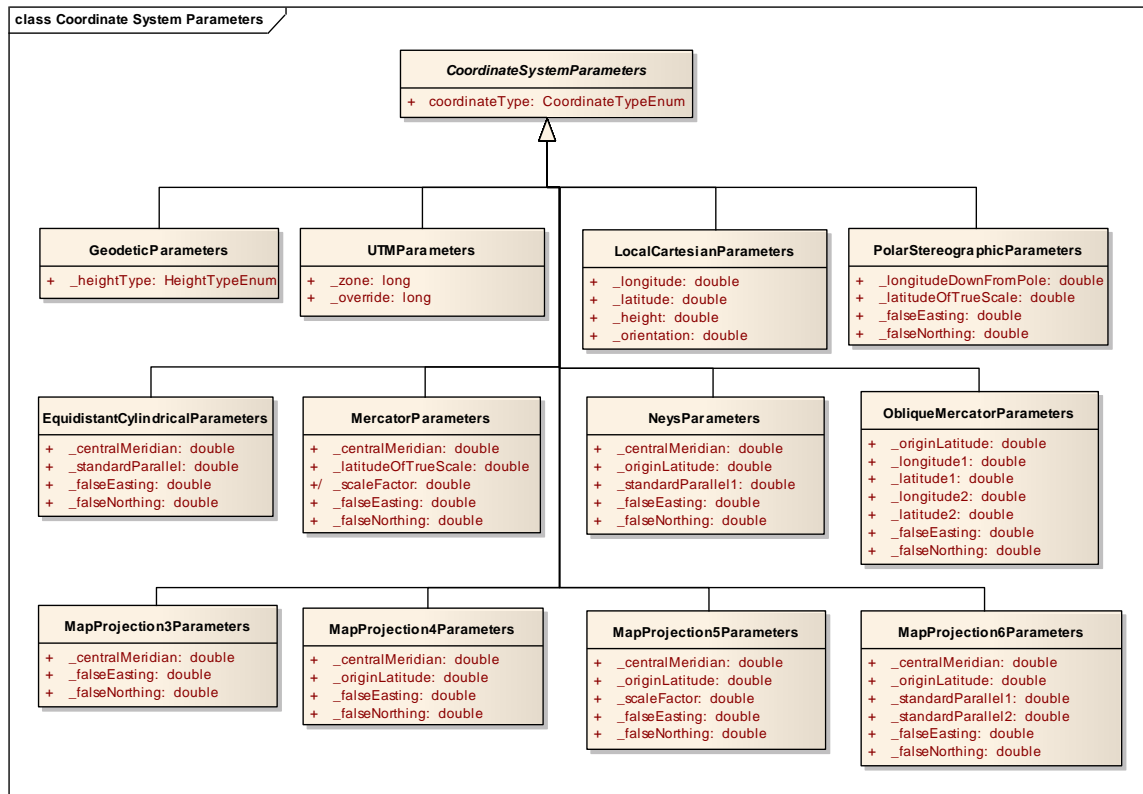
Figure 4-2. CoordinateConversionParameters and its Subclasses

- **dtcc/CoordinateSystems** – contains the source code for the abstract CoordinateSystem class and all of its concrete subclasses. It also contains classes for datums and ellipsoids. It is organized into subdirectories, as follows:

  - **albers** – contains the AlbersEqualAreaConic class, which performs conversions between geodetic coordinates and Albers Equal Area Conic projection coordinates.

  - **azeq** – contains the AzimuthalEquidistant class, which performs conversions between geodetic coordinates and Azimuthal Equidistant projection coordinates.

  - **bng** – contains the BritishNationalGrid class which, performs conversions between Geodetic and Transverse Mercator projection coordinates and British National Grid coordinates. It depends on the TransverseMercator class.

  - **bonne** – contains the Bonne class, which performs conversions between geodetic coordinates and Bonne projection coordinates.

  - **cassini**– contains the Cassini class, which performs conversions between geodetic coordinates and Cassini projection coordinates.

- **cyleqa** – contains the CylindricalEqualArea class, which performs conversions between Geodetic coordinates and Cylindrical Equal Area projection coordinates.

- **datum** – contains the DatumLibraryImplementation class, as well as the abstract Datum class and its concrete subclasses. These classes provide access to datum parameters for over 200 local datums, and perform datum transformations in either geodetic or geocentric coordinates.

- **eckert4**– contains the Eckert4 class, which performs conversions between geodetic coordinates and Eckert IV projection coordinates.

- **eckert6**– contains the Eckert6 class, which performs conversions between geodetic coordinates and Eckert VI projection coordinates.

- **ellipse** – contains the EllipsoidLibraryImplementation class, as well as the Ellipsoid class. These classes provide access to ellipsoid parameters for approximately two dozen standard ellipsoids.

- **eqdcyl**– contains the Equidistant Cylindrical class, which performs conversions between geodetic coordinates and Equidistant Cylindrical projection coordinates.

- **gars** – contains the GARS class, which performs conversions between geodetic coordinates and GARS coordinates.

- **geocent** – contains the Geocentric class, which performs conversions between geodetic coordinates and Geocentric coordinates.

- **georef** – contains the GEOREF class, which performs conversions between geodetic coordinates and GEOREF coordinates.

- **gnomonic** – contains the Gnomonic class, which performs conversions between geodetic coordinates and Gnomonic projection coordinates.

- **grinten**– contains the VanDerGrinten class, performs conversions between geodetic coordinates and Van der Grinten projection coordinates.

- **lambert** – contains the LambertConformalConic class, which performs conversions between geodetic coordinates and Lambert Conformal Conic projection coordinates.

- **loccart** – contains the LocalCartesian class, which performs conversions between geodetic and geocentric coordinates and Local Cartesian coordinates. It depends on the Geocentric class.

- **localspher** – contains the LocalSpherical class, which performs conversions between Local Spherical coordinates and Local Cartesian coordinates. It inherits from the LocalCartesian class.

- **mercator** – contains the Mercator class, which performs conversions between geodetic coordinates and Mercator projection coordinates.

- **mgrs** – contains the MGRS class, which performs conversions between Geodetic, UPS, and UPS coordinates and MGRS coordinates. It depends on the UPS and UTM classes.

- **miller**– contains the MillerCylindrical class, which performs conversions between geodetic coordinates and Miller Cylindrical projection coordinates.

- **misc**– contains the abstract CoordinateSystem class.

- **mollweid**– contains the Mollweide class, which performs conversions between geodetic coordinates and Mollweide projection coordinates.

- **neys** – contains the Neys class, which performs conversions between geodetic coordinates and Ney's (modified Lambert Conformal Conic) projection coordinates.

- **nzmg** – contains the NZMG class, which performs conversions between geodetic coordinates and New Zealand Map Grid (NZMG) coordinates.

- **omerc** – contains the ObliqueMercator class, which performs conversions between geodetic coordinates and Oblique Mercator projection coordinates.

- **orthog**– contains the Orthographic class, which performs conversions between geodetic coordinates and Orthographic projection coordinates.

- **polarst** – contains the PolarStereographic class, which performs conversions between geodetic coordinates and Polar Stereographic projection coordinates.

- **polycon**– contains the Polyconic class, which performs conversions between geodetic coordinates and Polyconic projection coordinates.

- **sinusoid**– contains the Sinusoidal class, which performs conversions between geodetic coordinates and Sinusoidal projection coordinates.

- **spherical**– contains the Spherical class, which performs conversions between geocentric coordinates and Spherical coordinates. It inherits from the Geocentric class.

- **stereogr** – contains the Stereographic class, which performs conversions between geodetic coordinates and Stereographic projection coordinates.

- **threads** – contains the Threads class.

- **tranmerc** – contains the TransverseMercator class, which performs conversions between geodetic coordinates and Transverse Mercator projection coordinates.

- **trcyleqa** – contains the TransverseCylindricalEqualArea class, which performs conversions between geodetic coordinates and Transverse Cylindrical Equal Area projection coordinates.

- **ups** – contains the UPS class, which performs conversions between geodetic coordinates and Universal Polar Stereographic projection coordinates. It depends on the POLAR STEREOGRAPHIC module.

- **usng** – contains the USNG class, which performs conversions between Geodetic, UPS, and UPS coordinates and USNG coordinates. It depends on the UPS and UTM class.

- **utm** – contains the UTM class, which performs conversions between geodetic coordinates and Universal Transverse Mercator projection coordinates. It depends on the TransverseMercator class.

- **webmercator** – contains the WebMercator class, which performs conversions from Web Mercator coordinates to Geodetic coordinates. The transform from Geodetic to Web Mercator is not supported since NGA wants to discourage the use of Web Mercator.

- **dtcc/CoordinateTuples** – contains the source code for the abstract CoordinateTuple class and all of its concrete subclasses, as well as the Accuracy class. These classes are shown in Figure 4-3.
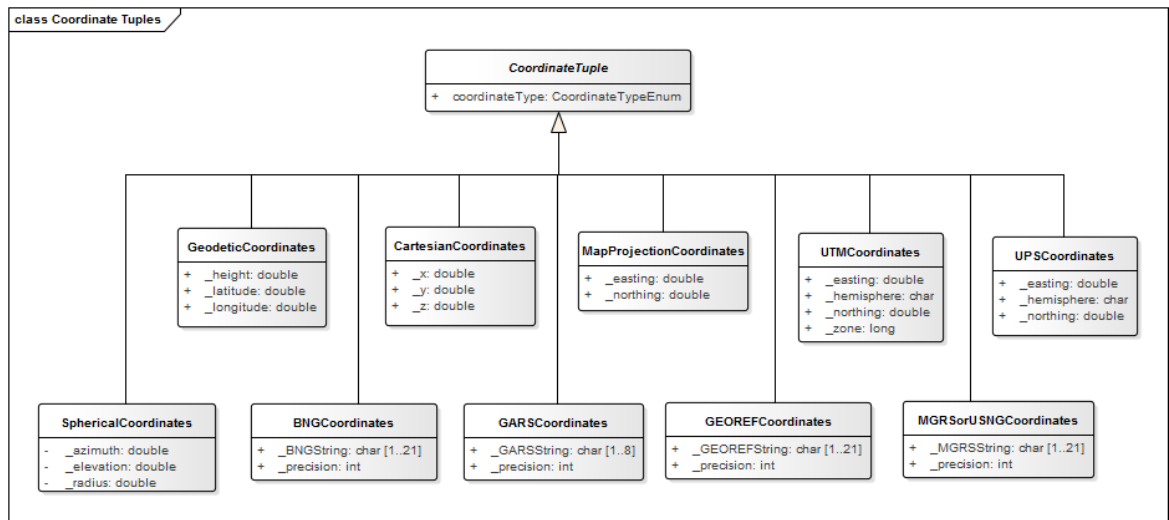
Figure 4-3. CoordinateTuple and its Subclasses

- **dtcc/Enumerations** – contains the various enumerations used by the CCS component. Note that the Precision enum meaning depends on the usage context. When used with a coordinate system expressed in degrees latitude and longitude its meaning is consistent with the enumeration values. When used with a coordinate system expressed in meters (e.g., MGRS) the meaning is metric units. For example, Precision enum value degree = 100km, tenMinute = 10km, and so forth down to tenThousandthOfSecond = 1mm.

- **dtcc/Exception** – contains the source code for the CoordinateConversionException class, as well as for the associated error and warning messages.

4.2 GEOTRANS APPLICATION SOFTWARE

This subsection describes the structure and organization of the GEOTRANS Application GUI software, which is located in the **GEOTRANS3** directory. It includes the GEOTRANS Application GUI source code, which is located in the **GEOTRANS3/java_gui/geotrans3** subdirectory, and the on-line help files, which are located in the **GEOTRANS3/help** subdirectory.

4.2.1 GEOTRANS APPLICATION GUI SOFTWARE

The GEOTRANS Application GUI software was generated using the NetBeans IDE, available at www.netbeans.org. The GEOTRANS Application GUI software is organized into subdirectories as follows:

- **coordinates** – contains the source code for the Java abstract CoordinateTuple class and all of its concrete subclasses, as well as the Accuracy and ConvertResults classes.

- **enumerations** – contains the source code for the Java classes that implement the various enumerations used by the GEOTRANS application. Note that the Precision enum meaning depends on the usage context. When used with a coordinate system expressed in degrees latitude and longitude, its meaning is consistent with the enumeration values. When used with a coordinate system expressed in meters (e.g., MGRS) the meaning is metric units. For example, Precision enum value degree = 100km, tenMinute = 10km, and so forth down to tenThousandthOfSecond = 1mm.

- **exception** – contains the source code for the Java classes that allow the application to handle exceptions thrown by the Coordinate Conversion Service.

- **gui** – contains the source code for the Java classes that implement the actual GUI components and controls, including the various control panels and dialog boxes.

- **jni** – contains the source code for the Java and C++ classes that implement a Java Native Interface (JNI) for the GEOTRANS Coordinate Conversion Service component.

- **misc** – contains a collection of miscellaneous Java support classes dealing with geodetic coordinate formatting options, conversion between character strings and coordinate values, loading and saving option settings, etc.

- **parameters** – contains the source code for the Java abstract CoordinateSystemParameters class and all of its concrete subclasses.

- **utility** – contains a collection of utility Java classes dealing with platforms, environment variables, directories and files, constants, and the on-line help browser interface.

4.2.2 ON-LINE HELP SOURCE FILES

This subsection describes the structure and organization of the On-Line Help source files, which are contained in the **GEOTRANS3/help** subdirectory. The source files are of two types. Files with the extension ".htm" are source files in Hyper Text Markup Language (HTML) which contain formatted text for one or more on-line help topics. Files with the extension ".gif" are graphics files, which contain the graphic images used in the on-line help topics. The On-Line Help source files consist of the following modules:

- Basic Operations (file **basicops.htm**) – describes the basic interactive operation of the GEOTRANS application.

- Contents (file **contents.htm**) – contains the top-level table of contents for the on-line help.

- Coordinate System Descriptions (file **coordDescriptions.htm**) – contains detailed descriptions of the various types of coordinate systems supported by the GEOTRANS application, including references to the following graphics:
  - Geodetic & Geocentric Coordinates (file **geodetic.gif**) – graphic showing the relationship between geodetic and geocentric coordinate systems (Figure 4.2 from TEC-SR-7).
  - Spherical Coordinates (file spherical.gif) – graphic shows the relationship between geocentric spherical and geocentric coordinate systems.
  - Local Spherical Coordinates (file locspher.gif) – graphic showing the relationship between local spherical and local cartesian coordinate systems.
  - GEOREF 15° Grid (file **georef1.gif**) – graphic showing the high-level 15° GEOREF grid.
  - GEOREF 1° Grid (file **georef2.gif**) – graphic showing the low-level 1° GEOREF grid.
  - MGRS Zones (file **mgrs_zones.gif**) – graphic showing the MGRS zones.
  - MGRS 100,000m Square Designations (AL Pattern) (file **mgrs_al.gif**) – graphic showing the patterns of MGRS 100,000m Square Designations used with the Clarke 1866, Clarke 1880, and Bessel 1841 ellipsoids.
  - MGRS 100,000m Square Designations (AA Pattern) (file **mgrs_aa.gif**) – graphic showing the patterns of MGRS 100,000m Square Designations used with all other ellipsoids.
  - MGRS 100,000m Square Designations for North Polar Zone (file **mgrs_n.gif**) – graphic showing the patterns of MGRS 100,000m Square Designations in the north polar zone.
  - MGRS 100,000m Square Designations for South Polar Zone (file **mgrs_s.gif**) – graphic showing the patterns of MGRS 100,000m Square Designations in the south polar zone.

- Coordinate Systems (file **coordsys.htm**) – contains brief descriptions of the various types of coordinate systems and map projections supported by the GEOTRANS application.

- Create Datum Dialog (file **createDatumDlg.htm**) – contains a description of the Create Datum dialog box.

- Create Ellipsoid Dialog (file **createEllipsoidDlg.htm**) – contains a description of the Create Ellipsoid dialog box.

- Datum Table (file **datum1.htm**) – contains a table of countries and geographic areas, identifying which local datums are applicable to each.

- Delete Datum Dialog (file **deleteDatumDlg.htm**) – contains a description of the Delete Datum dialog box.

- Delete Ellipsoid Dialog (file **deleteEllipsoidDlg.htm**) – contains a description of the Delete Ellipsoid dialog box.

- Ellipsoid Table (file **elliptab.htm**) – contains a table of standard ellipsoids and their parameters.

- Error Messages (file **errors.htm**) – contains descriptions of the error messages that may be output by the GEOTRANS application.

- Fields (file **fields.htm**) – contains descriptions of all of the fields and controls included in the graphical user interface of the GEOTRANS application.

- File Format (file **fileform.htm**) – contains a description of the format of the ASCII coordinate files read and created by the GEOTRANS application.

- File Format (file **fileHeader.htm**) – contains a description of the GEOTRANS Create File Header command.

- File Format (file **fileHeaderWindow.htm**) – contains a description of the Create File Header window.

- File Open Dialog (file **fileOpenDlg.htm**) – contains a description of the File Open dialog box.

- File Processing (file **fileproc.htm**) – describes how to use the file processing capabilities of the GEOTRANS application.

- File Processing Dialog (file **fileProcessingDlg.htm**) – contains a description of the File Processing dialog box.

- File Processing Window (file **fileProcessingWindow.htm**) – contains a description of the File Processing window.

- Glossary (file **glossary.htm**) – contains the definitions of general Mapping, Charting and Geodesy (MC&G) terms that are used in describing the GEOTRANS application.

- Main Window (file **mainWindow.htm**) – contains a description of the GEOTRANS application main window.

- Menu Commands (file **menubar.htm**) – contains descriptions of all of the menu commands provided by the menu bar in the GEOTRANS main window.

- Overview (file **overview.htm**) – contains an overview of the GEOTRANS application.

- Projection Descriptions (file **projectionDescriptions.htm**) – contains detailed descriptions of the various types of map projections supported by the GEOTRANS application, including references to the following graphics:
  - Mercator Projection (file **mercator.gif**) – graphic showing the Mercator projection.
  - Transverse Mercator Projection (file **transmerc.gif**) – graphic showing the Transverse Mercator projection.
  - UTM Projection (file **utm.gif**) – graphic showing meridians and parallels and a Universal Transverse Mercator grid (Figure 7.3 from TEC-SR-7).
  - Polar Stereographic Projection (file **polarst.gif**) – graphic showing meridians and parallels in a Polar Stereographic projection (Figure 7.4 from TEC-SR-7).
  - UPS Projection (file **ups.gif**) – graphic showing meridians and parallels on the Universal Polar Stereographic grid (Figure 7.6 from TEC-SR-7).
  - Albers Equal Area Conic projection (file **albers.gif**) – graphic showing an Albers Equal Area Conic projection.
  - Azimuthal Equidistant projection (file **azeq.gif**) – graphic showing an Azimuthal Equidistant projection.
  - Bonne projection (file **bonne.gif**) – graphic showing a Bonne projection.
  - British National Grid (file **bng.gif**) – graphic showing the British National Grid.
  - Cassini projection (file **cassini.gif**) – graphic showing a Cassini projection.

- Cylindrical Equal Area projection (file **cyleqa.gif**) – graphic showing a Cylindrical Equal Area projection.
- Eckert IV projection (file **eckert4.gif**) – graphic showing an Eckert IV projection.
- Eckert VI projection (file **eckert6.gif**) – graphic showing an Eckert VI projection.
- Equidistant Cylindrical projection (file **eqdcyl.gif**) – graphic showing an Equidistant Cylindrical projection.
- Gnomonic projection (file **gnomonic.gif**) – graphic showing a Gnomonic projection.
- Lambert Conformal Conic (1 parallel) Projection (file **lambert_1.gif**) – graphic showing a Lambert Conformal Conic (1 parallel) projection.
- Lambert Conformal Conic (2 parallel) Projection (file **lambert_2.gif**) – graphic showing a Lambert Conformal Conic (2 parallel) projection.
- Miller Cylindrical projection (file **miller.gif**) – graphic showing a Miller Cylindrical projection.
- Mollweide projection (file **mollweid.gif**) – graphic showing a Mollweide projection.
- Ney's Projection (file **neys71.gif**) – graphic showing Ney's projection.
- New Zealand Map Grid (file **nzmg.gif**) – graphic showing the New Zealand Map Grid.
- Oblique Mercator projection (file **omerc.gif**) – graphic showing an Oblique Mercator projection.
- Orthographic projection (file **ortho.gif**) – graphic showing a Orthographic projection.
- Polyconic projection (file **polyconic.gif**) – graphic showing a Polyconic projection.
- Sinusoidal projection (file **sinusoidal.gif**) – graphic showing a Sinusoidal projection.
- Stereographic projection (file **stereogr.gif**) – graphic showing a Stereographic projection.
- Transverse Cylindrical Equal Area projection (file **trcyleqa.gif**) – graphic showing a Transverse Cylindrical Equal Area projection.
- Van der Grinten projection (file **grinten.gif**) – graphic showing a Van der Grinten projection.

- Save As Dialog (file **saveAsDlg.htm**) – contains a description of the Save As dialog box.

4.3 DATA FILES

GEOTRANS requires several data files that contain datum, ellipsoid, and geoid separation information.   The following data files are normally located in the **data** directory, and GEOTRANS looks for them there by default:

- **3_param.dat** – the data file containing the datum shift parameters for 3-parameter local datums.

- **7_param.dat** – the data file containing the datum shift parameters for 7-parameter local datums.

- **Und_min2.5x2.5_egm2008_WGS84_TideFree_reformatted** – the data file containing the Earth Gravity Model 2008 (EGM2008) geoid separation values at 2.5-minute intervals.

- **egm96.grd** – the data file containing the Earth Gravity Model 1996 (EGM96) geoid separation values at 15-minute intervals.

- **egm84.grd** – the data file containing the Earth Gravity Model 1984 (EGM84) geoid separation values at 10-degree intervals.

- **wwgrid.bin** – the data file containing the Earth Gravity Model 1984 (EGM84) geoid separation values at 30-minute intervals.

- **ellips.dat** – the data file containing the standard ellipsoid parameters.

As an alternative, you can define the environment variable MSPCCS_DATA and place the files in the directory identified by this variable.

4.4 COMPILATION/BUILD PROCEDURES

The GEOTRANS executable directories: **GEOTRANS3/linux_64**, **GEOTRANS3/win** and **GEOTRANS3/win_64**, each contain required library files, executables for the GEOTRANS application, makefiles, and other supporting files.

4.4.1 LINUX COMPILATION/BUILD PROCEDURES

The CCS LINUX library directory, **CCS/linux_64**, contains the CCS shared objects and makefile.

To make the LINUX dtcc library, type:

> **`%make libMSPdtcc.so`**

To make the LINUX Coordinate Conversion Service library, type:

> **`%make libMSPCoordinateConversionService.so`**

The GEOTRANS LINUX executable directory, **`GEOTRANS3/linux_64`**, contains the executable, makefile, and supporting files.

To make the LINUX GEOTRANS Java Native Interface (JNI) shared object library, type:

> **`%make libjnimsp_ccs.so`**

To delete unneeded files created during the build process, type:

> **`%make clean`**

Note: The dtcc and Coordinate Conversion Service libraries should be built before trying to build the GEOTRANS JNI shared object library.

4.4.2 WINDOWS COMPILATION/BUILD PROCEDURES

The CCS Windows library directory, **CCS/win or CCS/win_64**, contains the CCS libraries and Microsoft Visual Studio 2015 solution and projects.

The GEOTRANS Windows executable directory, **`GEOTRANS3/win or GEOTRANS3/win_64`**, contains the GEOTRANS executable, Microsoft Visual Studio 2015 solution, projects, and supporting files.

To build the GEOTRANS Windows DLLs, start up the Microsoft Visual Studio 2015 and open the **`CCS/win/MSP_CCS.sln or CCS/win_64/MSP_CCS.sln`** solution file. In the **Solution Explorer** window, select the project to be built. The two projects are **MSP_DTCC**, which builds the dtcc DLL, and **MSP_CCS**, which builds the Coordinate Conversion Service DLL. Then, from the **`Build`** pop-up menu, select the **`Build`** command.

The C++ include directories for the **MSP_DTCC** project are as follows:

- ..\..\..\CCS\src\dtcc\CoordinateSystems\datum

- ../src/dtcc

- ../src/CoordinateConversion

- ../src/dtcc/CoordinateSystemParameters

- ../src/dtcc/CoordinateTuples

- ../src/dtcc/Enumerations

- ../src/dtcc/CoordinateSystems/misc

- ../src/dtcc/CoordinateSystems/datum

- ../src/dtcc/CoordinateSystems/ellipse

- ../src/dtcc/CoordinateSystems/albers

- ../src/dtcc/CoordinateSystems/azeq

- ../src/dtcc/CoordinateSystems/bng

- ../src/dtcc/CoordinateSystems/bonne

- ../src/dtcc/CoordinateSystems/cassini

- ../src/dtcc/CoordinateSystems/cyleqa

- ../src/dtcc/CoordinateSystems/eckert4

- ../src/dtcc/CoordinateSystems/eckert6

- ../src/dtcc/CoordinateSystems/eqdcyl

- ../src/dtcc/CoordinateSystems/gars

- ../src/dtcc/CoordinateSystems/geocent

- ../src/dtcc/CoordinateSystems/georef

- ../src/dtcc/CoordinateSystems/gnomonic

- ../src/dtcc/CoordinateSystems/grinten

- ../src/dtcc/CoordinateSystems/lambert

- ../src/dtcc/CoordinateSystems/loccart

- ../src/dtcc/CoordinateSystems/locspher

- ../src/dtcc/CoordinateSystems/Mercator

- ../src/dtcc/CoordinateSystems/mgrs

- ../src/dtcc/CoordinateSystems/miller

- ../src/dtcc/CoordinateSystems/mollweid

- ../src/dtcc/CoordinateSystems/neys

- ../src/dtcc/CoordinateSystems/nzmg

- ../src/dtcc/CoordinateSystems/omerc

- ../src/dtcc/CoordinateSystems/orthogr

- ../src/dtcc/CoordinateSystems/polarst

- ../src/dtcc/CoordinateSystems/polycon

- ../src/dtcc/CoordinateSystems/sinusoid

- ../src/dtcc/CoordinateSystems/spherical

- ../src/dtcc/CoordinateSystems/stereogr

- ../src/dtcc/CoordinateSystems/threads

- ../src/dtcc/CoordinateSystems/tranmerc

- ../src/dtcc/CoordinateSystems/trcyleqa

- ../src/dtcc/CoordinateSystems/ups

- ../src/dtcc/CoordinateSystems/usng

- ../src/dtcc/CoordinateSystems/utm

- ../src/dtcc/Exception.

The C++ include directories for the **MSP_CCS** project are as follows:

- ../src/dtcc

- ../src/CoordinateConversion

- ../src/dtcc/CoordinateSystemParameters

- ../src/dtcc/CoordinateTuples

- ../src/dtcc/Enumerations

- ../src/dtcc/CoordinateSystems/misc

- ../src/dtcc/CoordinateSystems/datum

- ../src/dtcc/CoordinateSystems/ellipse

- ../src/dtcc/CoordinateSystems/albers

- ../src/dtcc/CoordinateSystems/azeq

- ../src/dtcc/CoordinateSystems/bng

- ../src/dtcc/CoordinateSystems/bonne

- ../src/dtcc/CoordinateSystems/cassini

- ../src/dtcc/CoordinateSystems/cyleqa

- ../src/dtcc/CoordinateSystems/eckert4

- ../src/dtcc/CoordinateSystems/eckert6

- ../src/dtcc/CoordinateSystems/eqdcyl

- ../src/dtcc/CoordinateSystems/gars

- ../src/dtcc/CoordinateSystems/geocent

- ../src/dtcc/CoordinateSystems/georef

- ../src/dtcc/CoordinateSystems/gnomonic

- ../src/dtcc/CoordinateSystems/grinten

- ../src/dtcc/CoordinateSystems/lambert

- ../src/dtcc/CoordinateSystems/loccart

- ../src/dtcc/CoordinateSystems/locspher

- ../src/dtcc/CoordinateSystems/mercator

- ../src/dtcc/CoordinateSystems/mgrs

- ../src/dtcc/CoordinateSystems/miller

- ../src/dtcc/CoordinateSystems/mollweid

- ../src/dtcc/CoordinateSystems/neys

- ../src/dtcc/CoordinateSystems/nzmg

- ../src/dtcc/CoordinateSystems/omerc

- ../src/dtcc/CoordinateSystems/orthogr

- ../src/dtcc/CoordinateSystems/polarst

- ../src/dtcc/CoordinateSystems/polycon

- ../src/dtcc/CoordinateSystems/sinusoid

- ../src/dtcc/CoordinateSystems/spherical

- ../src/dtcc/CoordinateSystems/stereogr

- ../src/dtcc/CoordinateSystems/threads

- ../src/dtcc/CoordinateSystems/tranmerc

- ../src/dtcc/CoordinateSystems/trcyleqa

- ../src/dtcc/CoordinateSystems/ups

- ../src/dtcc/CoordinateSystems/usng

- ../src/dtcc/CoordinateSystems/utm

- ../src/dtcc/Exception

The GEOTRANS Java Native Interface DLL that supports the GEOTRANS Java GUI is built using the **jnigeotrans** solution. The one project is **jnigeotrans**, which builds the jnimsp_ccs DLL.

The C++ include directories for the jnigeotrans project are as follows:

- ..\..\..\..\include\win32

- ..\..\..\..\include

- ..\..\CCS\src\dtcc\CoordinateSystems\datum

- ..\..\CCS\src\dtcc\CoordinateSystems\ellipse

- ..\..\CCS\src\dtcc\Exception

- ..\..\CCS\src\CoordinateConversion

- ..\..\CCS\src\dtcc

- ..\..\CCS\src\dtcc\CoordinateSystemParameters

- ..\..\CCS\src\dtcc\CoordinateSystems\mercator

- ..\..\CCS\src\dtcc\CoordinateSystems\misc

- ..\..\CCS\src\dtcc\CoordinateTuples

- ..\..\CCS\src\dtcc\Enumerations

Document No. 2100501
15 April 2022

NOTE: Make sure to set environment variable MSPCCS_DATA when running in debug mode using the Visual Studio 2015 IDE. For example, in the MSPCCS_SpreadsheetTester project select Project->Properties. In the Property Pages window select debugging, then Environment. Enter the following text in the Environment window:

MSPCCS_DATA=..\..\data

PATH=..\..\CCS\win;.;%PATH%   REM 32-bit

PATH=..\..\CCS\win_64;.;%PATH%   REM 64-bit

4.4.3 JAVA COMPILATION/BUILD PROCEDURES

To compile and build the GEOTRANS Java files, start a NetBeans IDE, available at www.netbeans.org. Create a project for the **/GEOTRANS3/java_gui** directory in the filesystems explorer window by going to the **File** pull-down menu and selecting **New Project**. Select **Java** under **Categories** and **Java Project with Existing Sources** under **Projects**. Click the **Next** button. Type a **Project Name** and the **Project Folder** where you would like the project files to be. Under **Source Package Folders**, navigate to the **/GEOTRANS3/java_gui** directory and click the **Open** button. Click the **Next** button. Click the **Finish** button. Within the **Projects** window, right click on the project name. Select the **Build** command from the menu shown.

Next, edit the GEOTRANS Java archive (jar) file properties. First, click on the **Files** tab. Double click on Manifest.mf and insert the following line, **Main-Class: geotrans3/MSP_GEOTRANS3.** Next, click on nbproject under the GEOTRANS directory tree. Double click on project.properties. On the dist.jar line, type the directory and name of the jar file. The jar file is automatically created when the project is built.

5. NOTES

This section contains notes on aspects of the GEOTRANS software. Subsection 5.1 discusses the coordinate conversion logic that is implemented by the GEOTRANS Coordinate Conversion Service. Subsection 5.2 summarizes the datum transformation logic.

5.1 COORDINATE CONVERSION LOGIC

The function Convert in the GEOTRANS Coordinate Conversion Service component is the heart of the GEOTRANS software. It is this function that performs all coordinate conversions and datum shifts. Given the input and output datums, the input and output coordinate reference frames, the values of any projection parameters, and the input coordinates, it produces the output coordinates.

Although the source code for this function is quite long, its structure is relatively simple, consisting of nested if and switch statements. The code blocks that perform individual coordinate conversion and datum shift steps are quite short. Several special cases are handled initially to avoid unnecessary conversions.

The general case is handled in three stages:

1. Convert the input coordinates from the input coordinate reference frame to geodetic.

2. Shift the intermediate geodetic coordinates from the input datum to the output datum, converting between ellipsoid and geoid/MSL heights if necessary.

3. Convert the shifted intermediate geodetic coordinates to the output coordinate reference frame.

The first stage is accomplished by a switch statement on the input coordinate reference frame, with a case for each of the coordinate reference frames supported. If the input coordinates are already in geodetic, they are simply copied to the intermediate coordinates.

The second stage is accomplished in three steps. First, the intermediate geodetic coordinates are shifted to WGS 84, if necessary. Second, if the input geodetic coordinates included an ellipsoid height and the output geodetic coordinates require a geoid/MSL height, or vice versa, the geoid separation is either added to or subtracted from the height, as appropriate. Note that if either the input or the output was projected, the geodetic height is considered meaningless, and the output height is forced to zero. Third, the shifted intermediate geodetic coordinates are shifted to the output datum.

The third stage is similar to the first stage. A switch statement on the output coordinate reference frame, with a case for each of the coordinate reference frames supported, is used to convert the shifted intermediate geodetic coordinates to the output coordinate reference frame. If the output coordinate reference frame is geodetic, the shifted intermediate coordinates are simply copied to the output coordinates.

5.2 DATUM SHIFT LOGIC

Normally, a datum shift is performed in two stages: from the input datum to WGS 84, and then from WGS 84 to the output datum. If either the input datum or the output datum is WGS 84, then one or both of these stages will be skipped.

The methods used depend on the types of the input and output datums, and, in some cases, on the location of the coordinates being shifted. The datums are organized into the following categories:

- WGS 84
- WGS 72
- 3-parameter datums – local datums which are defined in the file **3-param.dat**, and which have three datum shift parameters (delta X, delta Y, delta Z).
- 7-parameter datums – local datums which are defined in the file **7-param.dat**, and which have seven datum shift parameters (delta X, delta Y, delta Z, X rotation, Y rotation, Z rotation, and scale factor).

Each of the 7-parameter datums (EUR-7 and OGB-7) also have 3-parameter versions (EUR-M and OGB-M). Distinct codes have been defined for these so that it is always clear which set of datum shift parameters should be used.

Four datum shift methods are used, as follows:

- 72-to-84 (and its inverse 84-to-72) – Used to shift between WGS 72 and WGS 84 in geodetic coordinates.
- Molodensky method – Used to shift between 3-parameter datums and WGS 84 in geodetic coordinates--except in the polar regions, where it is not valid.
- 3-parameter method – Used to shift between 3-parameter datums and WGS 84 in geocentric coordinates.
- 7-parameter method – Used to shift between 7-parameter datums and WGS 84 in geocentric coordinates.

APPENDIX A – ACRONYMS

API	Application Programming Interface
CCS	Coordinate Conversion Service
CD-ROM	Compact Disk – Read Only Memory
DLL	Dynamic Linked Library (Windows)
EGM	Earth Gravity Model
EPSG	European Petroleum Survey Group
GARS	Global Area Reference System
GCC	Gnu C++ Compiler
GEOREF	Geographic Reference System
GEOTRANS	Geographic Translator
GIF	Graphics Interchange Format
GUI	Graphical User Interface
HTML	HyperText Markup Language
IDE	Integrated Development Environment
JNI	Java Native Interface
LIB	Static Linked Library (Windows)
MGRS	Military Grid Reference System
MSL	Mean Sea Level
MSP	Mensuration Services Program
NGA	National Geospatial-Intelligence Agency
NZMG	New Zealand Map Grid
RHEL	Red Hat Enterprise Linux
UML	Universal Markup Language
UPS	Universal Polar Stereographic
USNG	United States National Grid

Document No. 2100501
15 April 2022

UTM   Universal Transverse Mercator

WGS   World Geodetic System

WWW   World Wide Web