Quest
Geo Solutions Ltd

Grid InQuest DLL User Manual

Version 6

Quest Geo Solutions Ltd
Passfield Business Center
Passfield
Hampshire, GU30 7SB

Tel	+44 (0) 1428 751 565
Fax	+44 (0) 1428 751 566

Licence Agreement

(For both personal and commercial use).

This is a legal agreement between you (either an individual or an entity), the end user, and Quest Geo Solutions Ltd. (Quest) regarding use of the software accompanying this Agreement ("Software"). By installing, copying, or otherwise using the Software, you agree to be bound by the terms of this Agreement. If you do not agree to the terms of this Agreement, do not install or use the Software.

The Software refers to the following program: Grid InQuest DLL (version 6)

The Software is provided to you free of charge. Should you accept the terms of this agreement, Quest will grant you the personal and non-exclusive right to install and use the software.

The Software or any other materials provided in this package may not be modified in any way. The Software may NOT be redistributed to friends or associates without the written permission of Quest. You may NOT upload the software to a freeware or shareware site without first receiving written permission from Quest.

No parts of The Software or other files provided in this package may be reproduced in part or in whole. You may not remove any copyright or other proprietary notice from the software. You may not reverse engineer, decompile, or disassemble the software.

Copyright Notice

The Software is Copyright (©) 2002-2004 by Quest Geo Solutions Ltd. The Software is protected by both the English copyright law and international copyright treaty provisions.

DISCLAIMER

The Software is provided "AS IS". You agree that by accepting this license, you are expressly acknowledging that the use of the software is AT YOUR OWN RISK.

QUEST GEO SOLUTIONS LTD. MAKES NO WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE SOFTWARE AND ANY ACCOMPANYING DOCUMENTATION.

IN NO EVENT SHALL QUEST GEO SOLUTIONS LTD. BE LIABLE FOR ANY DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF QUEST GEO SOLUTIONS LTD. HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Trademarks

Any product or brand names mentioned in the software and associated manuals and files are trademarks or registered trademarks of their respective owners.

YOU ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, UNDERSTAND IT, AND AGREE TO BE BOUND BY ITS TERMS AND CONDITIONS. YOU FURTHER AGREE THAT IT IS THE COMPLETE AND EXCLUSIVE STATEMENT OF THE AGREEMENT BETWEEN US THAT SUPERSEDES ANY PROPOSAL OR PRIOR AGREEMENT, ORAL OR WRITTEN, AND ANY OTHER COMMUNICATIONS BETWEEN US RELATING TO THE SUBJECT MATTER OF THIS AGREEMENT.

FIRST EDITION – 2002

Printed in the UK

Contents

1 Introduction

Welcome to the **Grid InQuest Dynamic Link Library (DLL)** manual. This document provides a complete reference for using and understanding the Grid InQuest library. It is intended to help you get started using **Grid InQuest** and to illustrate the methods and procedures involved in accurate coordinate conversion in Great Britain, Northern Ireland and the Republic of Ireland.

The manual is divided into three basic sections. The first section, ‘Getting Started’ teaches you the basics of the **Grid InQuest DLL** and a reference of all the library calls and return values. The second section, ‘Sample Code’ gives an example of using the dll in both a Visual C++ project and a Visual Basic project. The remainder of the manual deals with the technical aspects of the software, giving the theory behind the positioning algorithms.

1.1 Overview

The Grid InQuest ATL COM DLL provides a means for transforming coordinates between ETRS89 (WGS84) and the National coordinate systems of Great Britain, Northern Ireland and the Republic of Ireland. If necessary, it provides a fully three dimensional transformation incorporating the latest geoid model.

A coordinate may be set and retrieved as any of the following:

ETRS89 Cartesian
ETRS89 Geodetic
ETRS89 UTM
OSGB36
Irish Grid
Irish Transverse Mercator

In addition, depending on where the coordinates are located, it will convert between ETRS89 ellipsoidal heights and the following orthometric height datums:

Newlyn
St Marys (Scilly Isles)
Douglas02 (Isle of Man)
Stornoway (Outer Hebrides)
St Kilda
Lerwick (Shetland Isles)
Newlyn (Orkney Isles only)
Fair Isle
Flannan Isles
North Rona
Sule Skerry

Foula
Malin Head
Belfast Lough

1.2 System Requirements

The minimum software and hardware requirements to install and use **Grid InQuest DLL** are:

Hardware requirements:

IBM PC compatible with 80486 or higher processor (Pentium recommended)
64 MB of memory
50 MB of disk
CD drive

Operating System requirements:

Microsoft Windows 98 (Second Edition)
Microsoft Windows NT version 4.0 (SP3)
Microsoft Windows 2000
Microsoft Windows XP

1.3 Installation

To Install the Grid InQuest DLL

1. Insert the Grid InQuest DLL CD into your computer's CD-ROM drive. The Grid InQuest DLL Installation Wizard starts automatically. If the Wizard does not open automatically, you can use the `setup.exe` at the root of the CD instead.
2. Click **Next** to start the installation
3. In the **Select Installation Folder** screen you may choose a location to install the dll, however it is recommended that the default path is accepted. Click the *Browse* button to create or choose any folder on a hard disk with at least 8 megabytes of free disk space. Click **Next** to proceed to the next step
4. Click **Next** to start the installation, and the Wizard will start copying the required files to your chosen hard disk
5. Once the setup program has finished copying all the files, click *Restart Windows* if prompted to do so.

1.4 Accuracy of Ordnance Survey 2d Transformations

Within Great Britain, OSTN02 is the definitive OSGB36/ETRS89 transformation. OSTN02 in combination with the ETRS89 coordinates of the active GPS Network stations, rather than the fixed triangulation network, now define the National Grid. This means that, for example, the National Grid coordinates of an existing OSGB36 point, refixed using GPS from the National GPS Network and OSTN02, will be the correct ones. The original archived OSGB36 National Grid coordinates of the point (if different) will be wrong, by definition, but the two coordinates (new and archived) will agree on average to better than 0.1m (0.1m rmse, 68% probability).

Within the Republic of Ireland and Northern Ireland the OSi/OSNI polynomial transformation is recommended for coordinate transformations between ETRS89 and the Irish Grid. Transformed ETRS89 coordinates will agree with Irish Grid coordinates derived from traditional survey control to within 0.4m (95% data).

1.5 Accuracy of OSGM02

The heights output by precise GPS positioning in the ETRS89 coordinate system are geometric distance above the WGS84 (GRS80) reference ellipsoid. Note that GPS heights are typically two to three times less precise than horizontal positions. OSGM02 converts ETRS89 ellipsoidal heights to orthometric heights above mean sea level.

In mainland Great Britain, the datum (origin point) representing mean sea level is Ordnance Datum Newlyn, defined at Newlyn in Cornwall. In the Republic of Ireland, Northern Ireland, and the islands surrounding GB, mean sea level is defined by specific independent vertical datums which are all incorporated in OSGM02 and hence OSGM02 is compatible with the products from each of the Ordnance Surveys . Other Geoid models may give ‘mean sea level’ heights that are incompatible with the Ordnance Surveys products.

The estimated accuracies of OSGM02 for each regional vertical datum are included in the table below. The figures quoted assume precise ellipsoidal heights are used, for lower quality GPS observations additional error budget must be included.

Regional Datum	Standard Error (m)
Great Britain	0.02
Republic of Ireland	0.03
Northern Ireland	0.02
Orkney	0.08
Shetland	0.03
Outer Hebrides	0.09
Isle of Man	0.03
St. Kilda	0.06
Scilly Isles	Single offset from Newlyn

Any discrepancy found between an Ordnance Survey levelled bench mark (OSBM) and a OSGM02 computed orthometric height is likely to be due to bench mark subsidence or uplift and, assuming precise GPS survey has been carefully carried out, the orthometric height given by OSGM02 should be considered correct in preference to archive bench mark heights.

2 Getting Started

2.1 Overview

As part of the installation procedure, the dll should be registered on your computer. If it is not, it is normally installed under `\Windows\Quest`, and can be registered manually from the command prompt using the `regsvr32` command.

Essentially developers need to carry out the following steps in order to perform a coordinate transformation:

1. Load the dll into the software application
2. If the data file (GIQ60.dat) has been installed under a different location than the default, set the path of where the data file has been installed, either directly or through a registry key.
3. Set the area of interest (ie Great Britain, Northern Ireland or the Republic of Ireland)
4. Initialise the dll. Use the unlock string "`GIQ.6.0`" to pass into the function call. This will unlock the dll and load the transformation shifts and geoid model into memory.
5. Call one of the Set functions to set a coordinate in a particular format (eg ETRS89 Latitude, Longitude and ellipsoidal height, or OSGB36 Eastings, Northings and orthometric height).
6. Call one of the Get functions to retrieve the converted coordinate in a particular format (eg OSGB36 Eastings, Northings and orthometric height, or ETRS89 Latitude, Longitude and ellipsoidal height).
7. Repeat 5 and 6 as necessary.

2.2 Distributable Files

Once the application has been completed, the following files must be distributed as part of the Grid InQuest DLL:

1. GIQ60.dll
2. GIQ60.dat

Make sure the data file (GIQ60.dat) is installed in the same path as your software is expecting.

The remainder of the section gives a list of all the dll functionality that is available.

2.3 Set the path for the data files

Function Call:

```
eErrorCode SetDataFilesPath (BSTR bstrPath)
```

Inputs:	`BSTR`	Full path of the location of the data files
Return Values:	`eErrorCode`	
	`a) eSuccess`	All ok
	`b) eInvalidDataFilePath`	Data files not found under the path supplied

2.4 Set the path for the data files from a registry key

Function Call:

```
eErrorCode SetDataFilesPathFromReg (BSTR bstrRegKey)
```

Inputs:	`BSTR`	Full registry key containing the path of the location of the data files
Return Values:	`eErrorCode`	
	`a) eSuccess`	All ok
	`b) eInvalidRegKey`	Registry key not found
	`c) eInvalidDataFilePath`	Data files not found under the path supplied by the registry key

2.5 Set the area to work in

Function Call:

```
eErrorCode SetArea (eArea nArea)
```

Inputs:	`eArea`	Area to work in
	`a) eAreaGreatBritain`	Use the transformations for work in Great Britain
	`b) eAreaNI`	Use the transformations for work in Northern Ireland.
	`c) eAreaRoI`	Use the transformations for work in Republic of Ireland.

Return Values:	eErrorCode	
	a) eSuccess	All ok
	b) eFailure	Invalid area entered

2.6 Get the current working area

Function Call:

```
eErrorCode GetArea (eArea* pnArea)
```

Inputs:	eArea*	Pointer to the area
Outputs:	a) eAreaGreatBritain	Using the transformations for work in Great Britain
	b) eAreaNI	Using the transformations for work in Northern Ireland
	c) eAreaRoI	Using the transformations for work in Republic of Ireland.
Return Values:	eErrorCode	
	a) eSuccess	All ok

2.7 Initialise the DLL and load the data files

Function Call:

```
eErrorCode Initialise(BSTR bstrUnlockCode)
```

Inputs:	BSTR	DLL unlock string (“GIQ.6.0”)
Return Values:	eErrorCode	
	a) eSuccess	All ok
	b) eInvalidUnlockCode	The DLL unlock string supplied is invalid
	c) eInvalidDataFile	The data file can not be opened
	d) eAreaNotSet	No valid area has been set for the DLL to work with

2.8 Set an ETRS89 Cartesian coordinate

Function Call:

```
eErrorCode SetETRS89Cartesian(double dX, double dY, double dZ)
```

Inputs:	`double dX`	X Cartesian coordinate
	`double dY`	Y Cartesian coordinate
	`double dZ`	Z Cartesian coordinate

Return Values: `eErrorCode`

`a) eSuccess`	All ok
`b) eDLLNotInitialised`	The DLL has not been initialised
`c) eOutsideArea`	The supplied coordinate is outside the working area

2.9 Get an ETRS89 Cartesian coordinate

Function Call:

```
eErrorCode GetETRS89Cartesian(double* pdX, double* pdY, double*
                   pdZ)
```

Outputs:	`double* pdX`	pointer to the X Cartesian coordinate
	`double* pdY`	pointer to the Y Cartesian coordinate
	`double* pdZ`	pointer to the Z Cartesian coordinate

Return Values: `eErrorCode`

`a) eSuccess`	All ok
`b) eDLLNotInitialised`	The DLL has not been initialised

2.10 Set an ETRS89 Geodetic coordinate

Function Call:

```
eErrorCode  SetETRS89Geodetic(double  dLatDegs,  double  dLonDegs,
                              double dEllipHt)
```

Inputs:	`double dLatDegs`	ETRS89 Latitude in decimal degrees
	`double dLonDegs`	ETRS89 Longitude in decimal degrees
	`double dEllipHt`	ETRS89 Ellipsoidal height in metres

Return Values: `eErrorCode`

`a) eSuccess`	All ok
`b) eInvalidCoordinate`	An invalid coordinate has been supplied
`c) eDLLNotInitialised`	The DLL has not been initialised
`d) eOutsideArea`	The supplied coordinate is outside the working area

2.11 Get an ETRS89 Geodetic coordinate

Function Call:

```
eErrorCode GetETRS89Geodetic(double* pdLatDegs,
                             double* pdLonDegs,
                             double* pdEllipHt)
```

Inputs:	`double* pdLatDegs`	ETRS89 Latitude in decimal degrees
	`double* pdLonDegs`	ETRS89 Longitude in decimal degrees
	`double* pdEllipHt`	ETRS89 Ellipsoidal height in metres

Return Values: `eErrorCode`

`a) eSuccess`	All ok
`b) eDLLNotInitialised`	The DLL has not been initialised

2.12 Set an ETRS89 UTM coordinate

Function Call:

```
eErrorCode SetETRS89UTM( double dE, double dN, short nZone,
                         double dOrthoHt, eVertDatum* pnVertDatum)
```

Inputs:	`double dE`	UTM Eastings in metres on the ETRS89 datum
	`double dN`	UTM Northings in metres on the ETRS89 datum
	`short nZone`	UTM zone
	`double dOrthoHt`	Orthometric height in metres
	`eVertDatum* pnVertDatum`	Vertical datum to which the orthometric height is referred.

Return Values:	`eErrorCode`	
	`a) eSuccess`	All ok
	`b) eInvalidCoordinate`	An invalid coordinate has been supplied
	`c) eDLLNotInitialised`	The DLL has not been initialised
	`d) eOutsideArea`	The supplied coordinate is outside the working area

2.13 Get an ETRS89 UTM coordinate

Function Call:

```
eErrorCode GetETRS89UTM(double* pdE, double* pdN, short* pnZone
                      double* pdOrthoHt, eVertDatum* pnVertDatum)
```

Outputs:	`double* pdE`	UTM Eastings in metres on the ETRS89 datum
	`double* pdN`	UTM Northings in metres on the ETRS89 datum
	`short* pnZone`	UTM zone
	`double* pdOrthoHt`	Orthometric height in metres
	`eVertDatum* pnVertDatum`	Vertical datum to which the orthometric height is referred.

Return Values: eErrorCode

a) eSuccess	All ok
b) eInvalidCoordinate	An invalid coordinate has been supplied
c) eDLLNotInitialised	The DLL has not been initialised
d) eAreaNotSet	No valid area has been set for the DLL to work with

2.14 Set an Irish Grid coordinate

Function Call:

```
eErrorCode SetIrishGrid(double dE, double dN, double dOrthoHt,
                        eVertDatum nVertDatum)
```

Inputs:	double dE	Irish Grid Eastings in metres
	double dN	Irish Grid Northings in metres
	double dOrthoHt	Orthometric height in metres
	eVertDatum nVertDatum	Vertical datum to which the orthometric height is referred.

Return Values: eErrorCode

a) eSuccess	All ok
b) eInvalidCoordinate	An invalid coordinate has been supplied
c) eInvalidVertDatum	An invalid vertical datum has been supplied
d) eDLLNotInitialised	The DLL has not been initialised
e) eOutsideArea	The supplied coordinate is outside the working area

2.15 Get an Irish Grid coordinate

Function Call:

```
eErrorCode GetIrishGrid(double* pdE, double* pdN,
                        double* pdOrthoHt,
                        eVertDatum* pnVertDatum)
```

Inputs:	eVertDatum* pnVertDatum	Vertical datum to which the orthometric height is to be referred.

Outputs:	`double* pdE`	Irish Grid Eastings in metres
	`double* pdN`	Irish Grid Northings in metres
	`double* pdOrthoHt`	Orthometric height in metres
	`eVertDatum* pnVertDatum`	Vertical datum to which the orthometric height is referred.
Return Values:	`eErrorCode`	
	a) `eSuccess`	All ok
	b) `eInvalidCoordinate`	An invalid coordinate has been supplied
	c) `eInvalidVertDatum`	An invalid vertical datum has been supplied
	d) `eDLLNotInitialised`	The DLL has not been initialised
	e) `eAreaNotSet`	No valid area has been set for the DLL to work with

2.16 Set an Irish Transverse Mercator coordinate

Function Call:

```
eErrorCode SetIrishTransMerc(double dE, double dN,
                             double dOrthoHt,
                             eVertDatum nVertDatum)
```

Inputs:	`double dE`	Irish Transverse Mercator Eastings in metres
	`double dN`	Irish Transverse Mercator Northings in metres
	`double dOrthoHt`	Orthometric height in metres
	`eVertDatum nVertDatum`	Vertical datum to which the orthometric height is referred.
Return Values:	`eErrorCode`	
	a) `eSuccess`	All ok
	b) `eInvalidCoordinate`	An invalid coordinate has been supplied
	c) `eInvalidVertDatum`	An invalid vertical datum has been supplied
	d) `eDLLNotInitialised`	The DLL has not been initialised
	e) `eOutsideArea`	The supplied coordinate is outside the working area

2.17 Get an Irish Transverse Mercator coordinate

Function Call:

```
eErrorCode GetIrishTransMerc(double* pdE, double* pdN,
                             double* pdOrthoHt,
                             eVertDatum* pnVertDatum)
```

Inputs:	`eVertDatum* pnVertDatum`	Vertical datum to which the orthometric height is to be referred.
Outputs:	`double* pdE`	Irish Transverse Mercator Eastings in metres
	`double* pdN`	Irish Transverse Mercator Northings in metres
	`double* pdOrthoHt`	Orthometric height in metres
	`eVertDatum* pnVertDatum`	Vertical datum to which the orthometric height is referred.
Return Values:	`eErrorCode`	
	a) `eSuccess`	All ok
	b) `eInvalidCoordinate`	An invalid coordinate has been supplied
	c) `eInvalidVertDatum`	An invalid vertical datum has been supplied
	d) `eDLLNotInitialised`	The DLL has not been initialised
	e) `eAreaNotSet`	No valid area has been set for the DLL to work with

2.18 Set an OSGB36 coordinate

Function Call:

```
eErrorCode SetOSGB36(double dE, double dN, double dOrthoHt,
                     eVertDatum* pnVertDatum)
```

Inputs:	`double dE`	OSGB36 Eastings in metres
	`double dN`	OSGB36 Northings in metres
	`double dOrthoHt`	Orthometric height in metres
	`eVertDatum* pnVertDatum`	Vertical datum to which the orthometric height is referred. This variable will be changed to the correct

vertical datum for the coordinate supplied, if the incorrect vertical datum is entered.

Return Values: `eErrorCode`

a) `eSuccess`	All ok
b) `eInvalidVertDatum`	An invalid vertical datum has been supplied
c) `eDLLNotInitialised`	The DLL has not been initialised
d) `eOutsideArea`	The supplied coordinate is outside the working area

2.19 Get an OSGB36 coordinate

Function Call:

```
eErrorCode GetOSGB36(double* pdE, double* pdN,double* pdOrthoHt,
                     eVertDatum* pnVertDatum)
```

Inputs:	`eVertDatum* pnVertDatum`	Vertical datum to which the orthometric height is to be referred.

Outputs:	`double* pdE`	OSGB36 Eastings in metres
	`double* pdN`	OSGB36 Northings in metres
	`double* pdOrthoHt`	Orthometric height in metres
	`eVertDatum* pnVertDatum`	Vertical datum to which the orthometric height is referred.

Return Values: `eErrorCode`

a) `eSuccess`	All ok
b) `eInvalidVertDatum`	An invalid vertical datum has been supplied
c) `eDLLNotInitialised`	The DLL has not been initialised
d) `eAreaNotSet`	No valid area has been set for the DLL to work with

2.20 DLL Constant Definitions

Areas

eArea

Constant	Value
eAreaGreatBritain	0
eAreaNI	1
eAreaRoI	2
eAreaUnknown	-1

Error Codes

eErrorCode

Constant	Value
eSuccess	0
eFailure	-1
eInvalidDataFilePath	-2
eInvalidRegKey	-3
eInvalidUnlockCode	-4
eInvalidDataFile	-5
eInvalidPolygonFile	-6
eAreaNotSet	-7
eDLLNotInitialised	-8
eInvalidCoordinate	-9
eInvalidVertDatum	-10
eOutsideArea	-11

Vertical Datums

eVertDatum

Constant	Value
eUnknownVertDatum	0
eNewlynDatum	1
eScillyIslesDatum	2
eIsleManDatum	3
eOuterHebridesDatum	4
eStKildaDatum	5
eShetlandDatum	6
eOrkneyDatum	7
eFairIsleDatum	8
eFlannansDatum	9
eNorthRonaDatum	10
eSuleSkerryDatum	11
eFoulaDatum	12
eMalinHeadDatum	13
eBelfastDatum	14

3 Sample Code

3.1 Sample C++ Code

Incorporate the information from the library by using the `#import` directive:

```
#import “\Windows\Quest\GIQ60.dll” no_namespace named_guids
```

This will then give access to the `IOSTransformationPtr` interface, on which an instance of the COM object `OSTransformation` can be created.

For example:

```
IOSTransformationPtr pOSTrans = NULL;

if(SUCCEEDED(pOSTrans.CreateInstance(__uuidof(OSTransformation))))
{
     ….
}
```

The pointer to the interface may then be used to access the libraries function calls and procedures.

By default, the setup program will install both the dll and the data files to `\Windows\Quest` (or `\WinNT\Quest`). Certain applications may require the data files to be located in different directories. The path for the data files may be set directly:

```
if(pOSTrans->SetDataFilesPath("C:\\Raw Data") != eSuccess)
{
     AfxMessageBox("Unable to set data files path");
}
```

or from a key in the registry containing the path name:

```
if(pOSTrans->SetDataFilesPathFromReg("HKEY_CURRENT_USER\\Software\\Quest")
                                                != eSuccess)
{
     AfxMessageBox("Unable to set data files path");
}
```

The area in which coordinates are to be transformed can then be set, and the DLL unlocked using the unlock string:

```
if(pOSTrans->SetArea(eAreaGreatBritain) != eSuccess)
{
      AfxMessageBox("ERROR: Trying to set an invalid area!");
}

// --- Unlock the DLL and load the data files into memory --- //
nRes = pOSTrans->Initialise("GIQ.6.0");
```

The DLL is now set up to provide all the necessary transformation procedures. For example, to convert between ETRS89 Cartesian and OSGB36:

```
// --- Set the coordinate as ETRS89 Cartesian --- //
if(pOSTrans->SetETRS89Cartesian(3875266.764,116103.381,5047549.096)
                       != eSuccess)
{
      AfxMessageBox("ERROR: See error code from return value");
}
if(pOSTrans->GetOSGB36(&dE,&dN,&dOrthoHt,&nVertDatum) != eSuccess)
{
      AfxMessageBox("ERROR: See error code from return value");
}
```

This sets the Cartesian coordinates (3875266.764 mX,116103.381 mY, 5047549.096 mZ) and then retrieves the corresponding OSGB36 Eastings and Northings in the variables `dE` and `dN`, together with the orthometric height in the variable `dOrthoHt`, and the vertical datum code in the variable `nVertDatum` (for example, `eNewlynDatum`).

3.2 Sample VB Code

Incorporate the dll into the VB project by selecting *Project* from the menu, and then selecting *References*. A dialog box listing all the available registered dlls and COM objects will be displayed. Scroll down the list, and select:

GIQ60 1.0 Type Library

In the form or module that you want to use the dll functionality, create an object as follows:

```
Public osTransform As New GIQ60Lib.OSTransformation
```

The variable `osTransform` can be used to interface to access the libraries function calls and procedures.

By default, the setup program will install both the dll and the data files to `\Windows\Quest` (or `\WinNT\Quest`). Certain applications may require the data files to be located in different directories. The path for the data files may be set directly:

```
If Not osTransform.SetDataFilesPath("C:\Raw Data") =
            GIQ60Lib.eErrorCode.eSuccess Then
      MsgBox("Unable to set data files path")
End If
```

or from a key in the registry containing the path name:

```
If Not osTransform.SetDataFilesPathFromReg("HKEY_CURRENT_USER\Software\Quest")
            = GIQ60Lib. eErrorCode.eSuccess) Then
      MsgBox("Unable to set data files path")
End If
```

The area in which coordinates are to be transformed can then be set, and the DLL unlocked using the unlock string:

```
If Not osTransform.SetArea(eAreaGreatBritain) =
            GIQ60Lib.eErrorCode.eSuccess) Then
      MsgBox("ERROR: Trying to set an invalid area!");
End If

' Unlock the DLL and load the data files into memory
nRes = osTransform.Initialise("GIQ.6.0")
```

The DLL is now set up to provide all the necessary transformation procedures. For example, to convert between ETRS89 Cartesian and OSGB36:

```
‘ Set the coordinate as ETRS89 Cartesian
If Not osTransform.SetETRS89Cartesian(3875266.764,116103.381,5047549.096)
                        = GIQ60Lib.eErrorCode.eSuccess) Then
      MsgBox("ERROR: See error code from return value")
End If

Dim east As Double = 0, north As Double = 0, ht As Double = 0
Dim pnDatum As GIQ60Lib.eVertDatum

If Not osTransform.GetOSGB36(east, north, ht, pnDatum) =
            GIQ60Lib.eErrorCode.eSuccess Then
      MsgBox("ERROR: See error code from return value")
End If
```

This sets the Cartesian coordinates (3875266.764 mX,116103.381 mY, 5047549.096 mZ) and then retrieves the corresponding OSGB36 Eastings and Northings in the variables `dE` and `dN`, together with the orthometric height in the variable `dOrthoHt`, and the vertical datum code in the variable `nVertDatum` (for example, `eNewlynDatum`).

4 About the Transformations

All Ordnance Survey mapping relates to a coordinate reference system. In Great Britain OSGB coordinates relate to OSGB36® (the National Grid), within Northern Ireland and the Republic of Ireland the coordinate reference system is the Irish Grid. These reference systems are traditionally realised on the earth's surface by monumented triangulation stations. The users of mapping products, in both the public and private sectors, have invested in geographical information systems (GIS) and asset management systems based on these Grid systems which have been accepted as de facto national standards.

In order to relate GPS-derived positions to the Ordnance Surveys' mapping, GPS derived coordinates need to be converted to Irish Grid or to National Grid, which requires a specialised datum transformation. For this reason the Ordnance Survey of Northern Ireland and Ordnance Survey Ireland have developed a polynomial transformation, which is the standard datum transformation for use throughout Ireland. The Ordnance Survey of Great Britain have developed OSTN02, the standard datum transformation for Great Britain.

Ordnance Survey mapping also includes height information that relates to a regional vertical datum. Height information in Great Britain refers to Ordnance Datum Newlyn (ODN), which is established from mean sea level. Although ODN is the national height datum used across mainland Great Britain there are a number of additional datums that are used on the surrounding islands, namely: Lerwick on the Shetland Islands; Stornoway on the Outer Hebrides; St. Kilda; Douglas02 on the Isle of Man and St. Marys on the Scilly Isles. The Ordnance Survey of Northern Ireland relates heights within Northern Ireland to Belfast Lough datum, and Ordnance Survey Ireland relates heights within the Republic of Ireland to the Malin Head datum. The resulting Ordnance Survey Geoid model (OSGM02) incorporates all the above vertical datums.

4.1 OSTN02

The Ordnance Survey of Great Britain has developed the horizontal transformation OSTN02. This transformation consists of a 1,250km by 700km grid of translation vectors at 1km resolution. This provides a fit between the GPS coordinate system ETRS89 and the OSGB36 National Grid. ETRS89 coordinates can be determined in Great Britain by linking a GPS survey to Active or Passive stations from the British 'National GPS Network', see www.gps.gov.uk. OSTN02 is in agreement with major triangulation stations at the level of 0.1m root mean square (RMSE).

Within Great Britain OSTN02 (the Ordnance Survey National Grid Transformation), in conjunction with the ETRS89 positions of the active GPS Network stations, is now the official definition of OSGB36 National Grid coordinate system. This means that using OSTN02 with the National GPS Network, surveyors using GPS have no need to occupy triangulation stations in order to relate GPS coordinates to National Grid coordinates.

4.2 OSi/OSNI Polynomial Transformation

Ordnance Survey Ireland and Ordnance Survey of Northern Ireland recommend the OS*i*/OSNI polynomial transformation for all horizontal transformations in the Republic of Ireland and Northern Ireland. This transformation has been developed in association with the Institute of Engineering Surveying and Space Geodesy, University of Nottingham.

The transformation is based on 183 points evenly distributed throughout Ireland and Northern Ireland. The precise ETRS89 and Irish Grid coordinates of these points ere determined by GPS and terrestrial survey methods, and a one-dimensional 3rd order polynomial individually fitted to the latitude and the longitude. The resulting polynomial allows calculation of the coordinate differences at additional points. The polynomial transformation has an accuracy of 0.4m (95% data).

4.3 Ordnance Survey Geoid Model: OSGM02

To provide the third dimension of the transformation, the Ordnance Surveys have, with others, developed the Geoid model OSGM02. The model is derived from precise gravity surveys across UK, Ireland, and surrounding waters, additionally the model includes data from the global geopotential model (EGM96). Alignment to each regional vertical datum is based on precise GPS observations at Ordnance Survey levelling marks. Within Great Britain these include the Ordnance Survey fundamental benchmark network

The Geoid model consists of a 1km grid with geoid-ellipsoid separation values covering all of Great Britain, Ireland and Northern Ireland. This model can be used with GPS determined positions to establish height above mean sea level, as defined by the respective vertical datums, to the accuracies shown in the table below. The Ordnance Surveys recommend the use of the Geoid Model OSGM02 and the active GPS network to produce orthometric height compatible with Ordnance Survey mapping.

> OSGB intend that OSGM02 is the official definition of the relationship between GPS ellipsoid heights and orthometric height in Great Britain. In the way that GPS and the transformation model OSTN02 define the horizontal coordinate system, precise GPS surveying using the OSGB active GPS Network in conjunction with the Geoid model will become the standard method of determining orthometric height.

OSTN02 covers Great Britain and the Isle of Man. The OS*i*/OSNI polynomial transformation covers the Republic of Ireland and Northern Ireland. It should be noted that the Irish Grid and the National Grid are two independent coordinate reference systems, and that Irish Grid coordinates are not directly compatible with OSGB36 coordinates.

OSGM02 covers all of Great Britain, Isle of Man, Republic of Ireland, and Northern

Ireland. The Geoid model comprises of 14 patches in order to relate to mean sea level as defined by the specific vertical datum for each region. The Datum Flag which forms part of each data record specifies to which datum the geoid/ellipsoid separation value relates.

Both models have been ‘cookie-cut’ to a boundary which extends 10km offshore. Any point outside this boundary will return null values in the shift and datum flag records. It is strongly suggested that any software written to incorporate this data be capable of recognising a null value and to return an ‘outside of model boundary’ error message.

Within Ireland and Northern Ireland, OSGM02 returns orthometric heights relative to the Malin Head and Belfast Lough datums respectively. OSGM02 will return orthometric height relative to either the Malin Head or the Belfast Lough datums for points within 2km of the border between the Republic of Ireland and Northern Ireland. It is recommended that software written to incorporate this data does not extend the model beyond these limits.