Transform Panel Feature Coordinate With Transformation Assistant API

Caleb Palmer

Exported on Feb 16, 2021

Table of Contents

1 Prerequisites 4

2 Step-by-step guide 5

3 Sample C#.NET Project 15

4 Related articles 16

The coordinates for features in a Paper Chart are stored as CHMR coordinates and cannot be transformed to LLDG by the HPD Server API alone.  This demokit will describe the steps needed to get the coordinates of a feature from a Paper Chart Panel using the HPD Server API and converting them to LLDG coordinates using the Transformation Assistant API.  It is accompanied by a sample C# solution which provides a command line program for retrieving the LLDG coordinates of an $rncpanel feature for a given Paper Chart Panel.

# Prerequisites

1. An Oracle Client installed
2. Microsoft Visual Studio with C#

# Step-by-step guide

1. In Microsoft Visual Studio, create a new C# project.
2. Add a new class called TransformationAssistant.  This class will provide the interface to the transformationassistantu.dll which is shipped with HPD Suite.  The documention for this API is provided in the CARIS HPD PCE API Reference Guide under the section "Transforming Coordinates".
3. Add the following to the class:

|  |
| --- |
| public class TransformationAssistant  {  private const string dllPath = @"C:\Program Files\CARIS\HPD Suite\3.1\Bin\transformationassistantu.dll";    [DllImport(dllPath,  EntryPoint = @"?initializeEnvironment@Utilities@TransformationAssistant@Caris@@YAJPEBD@Z")]  public static extern int initializeEnvironment(string datumPath);    [DllImport(dllPath,  EntryPoint = "?transformToProductLLDG@CHMRToLLDG@TransformationAssistant@Caris@@YAHQEA\_W00NNNNNNNNNNNNNNNNNNNNNNAEAN1@Z",  CharSet = CharSet.Unicode,  CallingConvention = CallingConvention.Cdecl)]  public static extern Int32 transformToProductLLDG(  string panelEllipsoidName,  string sourceEllipsoidName,  string projectionInfo,  double scale,  double originX,  double originY,  double datumSemiMajorAxis,  double datumSemiMinorAxis,  double datumX0,  double datumY0,  double datumZ0,  double datumRx,  double datumRy,  double datumRz,  double datumSf,  double datumXShift,  double datumYShift,  double datumLatShift,  double datumLonShift,  double alignX1,  double alignY1,  double alignX2,  double alignY2,  double inXToBeTransformed,  double inYToBeTransformed,  ref double outTransformedLongitude,  ref double outTransformedLatitude);    [DllImport(dllPath,  EntryPoint = "?transformToProductLLDG@CHMRToLLDG@TransformationAssistant@Caris@@YAHQEA\_W00NNNNNNNNNAEAN1@Z",  CharSet = CharSet.Unicode,  CallingConvention = CallingConvention.Cdecl)]  public static extern Int32 transformToProductLLDG(  string panelEllipsoidName,  string sourceEllipsoidName,  string projectionInfo,  double scale,  double originX,  double originY,  double alignX1,  double alignY1,  double alignX2,  double alignY2,  double inXToBeTransformed,  double inYToBeTransformed,  ref double outTransformedLongitude,  ref double outTransformedLatitude);    [DllImport(dllPath,  EntryPoint = "?getErrorMessage@Utilities@TransformationAssistant@Caris@@YAPEA\_WJ@Z")]  public static extern String getErrorMessage(int errorcode);  } |

Code Block 1 TransformationAssistant.cs

Line 3 defines the absolute path to the DLL which exports the functions that will be used to perform a CHMR to LLDG coordinate transformation.  Change this to the path the is appropriate for the HPD Suite Installation that will be used.Lines 6, 10, 43, and 63 provide the entry point locations into transformationassistantu.dll for the exported functions that will be used.  The provided entry points should be updated with the appropriate entry points for the environment this is being used in.  Open a command line and use the dumpbin program to list out the entry points for the functions that are exported by this DLL:

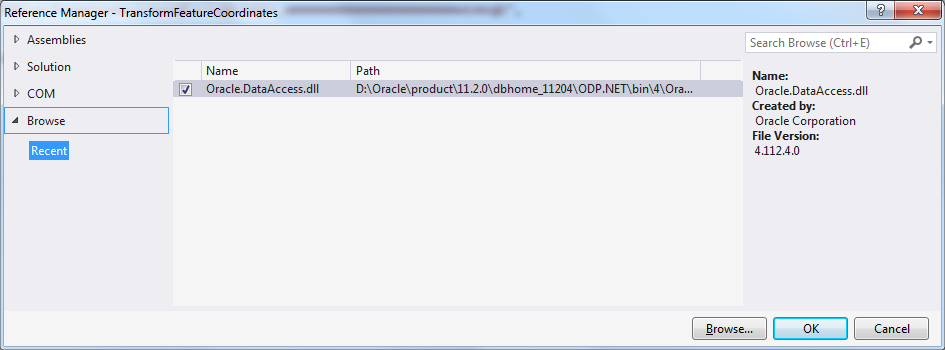
|  |
| --- |
| dumpbin /exports "C:\Program Files\CARIS\HPD Suite\3.1\Bin\transformationassistantu.dll" |

It will output something like this:

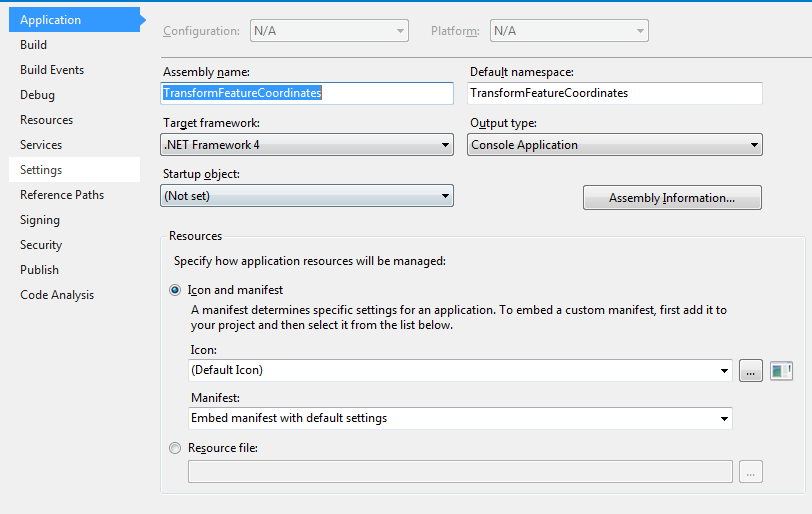
|  |
| --- |
| Dump of file C:\Program Files\CARIS\HPD Suite\3.1\Bin\transformationassistantu.dll File Type: DLL  Section contains the following exports for transformationassistantu.dll  00000000 characteristics  54F5D44D time date stamp Tue Mar 03 11:33:33 2015  0.00 version  1 ordinal base  6 number of functions  6 number of names  ordinal hint RVA name  1 0 00003EC0 ?getErrorMessage@Utilities@TransformationAssistant@Caris@@YAPEA\_WJ@Z  2 1 00004060 ?initializeEnvironment@Utilities@TransformationAssistant@Caris@@YAJPEBD@Z  3 2 00002C80 ?transformToProductLLDG@CHMRToLLDG@TransformationAssistant@Caris@@YAHQEA\_W00NNNNNNNNNAEAN1@Z  4 3 00002800 ?transformToProductLLDG@CHMRToLLDG@TransformationAssistant@Caris@@YAHQEA\_W00NNNNNNNNNNNNNNNNNNNNNNAEAN1@Z  5 4 00002460 ?transformToSourceLLDG@CHMRToLLDG@TransformationAssistant@Caris@@YAHQEA\_W00NNNNNNNNNAEAN1@Z  6 5 00001F20 ?transformToSourceLLDG@CHMRToLLDG@TransformationAssistant@Caris@@YAHQEA\_W00NNNNNNNNNNNNNNNNNNNNNNAEAN1@Z  Summary  2000 .data  1000 .pdata  7000 .rdata  1000 .reloc  1000 .rsrc |

Update lines 6, 10, 42, and 63 with the appropriate Entry Point values output by dumpbin.

1. Create a new class which will contain the main functionality called Program.cs.
2. This class will use the HPD Server API so it will need to use the Oracle ODP.Net assembly which is provided with the Oracle client.  Right-Click on References in Solution Explorer, and click Add-Reference.  Click on the Browse tab and then click the browse button.  In the file selection dialog, navigate to the ORACLE\_HOME of the Oracle client and select this file:  *ORACLE\_HOME\*ODP.NET\bin\4\Oracle.DataAccess.dll.  Click Add.  The Reference Manager window should still be open.  Ensure that Oracle.DataAccess.dll is checked and then click OK.



1. Change the Projects target framework to .NET 4.0 to match the Oracle.DataAccess.dll.  Right-click on the project in Solution Explorer, under the application tab, set Target Framework to .NET 4.0.



|  |
| --- |
| There are a number of steps to preparing the HPD connection, getting the geometry, and transforming the geometry to LLDG.  They are as follows:   1. Create the Oracle Connection 2. Set the current schema of the connection 3. Populate the temporary tables based on the Feature and the Chart Panel of interest. 4. Get the Geometry using the p\_pprapi.GetGeometry() function. 5. Prepare for the coordinate system transformation by getting the necessary coordinate system parameters   The following steps will describe this. |

1. Add the following lines at the top of the file to import the Oracle.DataAccess assembly:

|  |
| --- |
| using Oracle.DataAccess.Client;  using Oracle.DataAccess.Types; |

Add the following lines in your main function (Or another function) to create the Oracle connection:

|  |
| --- |
| string connString = System.String.Format("Data Source={0};User ID={1};Password={2}", servicename, username, password);  OracleConnection oraConn = new OracleConnection(connString);  oraConn.Open(); |

Update the servicename, username, and password arguments as appropriate.

1. Change the the current schema and NLS settings of the connected session to the HPD schema:

|  |
| --- |
| string sql = @"BEGIN  execute immediate 'alter session set current\_schema = ' || p\_schema\_constants.returnschema();  execute immediate 'alter session set NLS\_LANGUAGE = AMERICAN';  execute immediate 'alter session set NLS\_TERRITORY = AMERICA';  END;  ";  OracleCommand oraCmd = new OracleCommand(sql, oraConn);  oraCmd.ExecuteNonQuery();  oraCmd.Dispose(); |

1. Populate the temporary tables:

|  |
| --- |
| // Populate tempfoids  string sql = "DELETE FROM tempfoids";  OracleCommand oraCmd = new OracleCommand(sql, oraConn);  oraCmd.ExecuteNonQuery();  oraCmd.Dispose();    sql = "insert into TEMPFOIDS (select foid from panel\_feature\_vw where panelver\_id = :panelversion\_id)";  oraCmd = new OracleCommand(sql, oraConn);  OracleParameter oraParam = new OracleParameter();  oraParam.OracleDbType = OracleDbType.Int32;  oraParam.Value = panelVersionId;  oraCmd.Parameters.Add(oraParam);  oraCmd.ExecuteNonQuery();  oraParam.Dispose();  oraCmd.Dispose();    // Populate tempfeatureversion  sql = "DELETE FROM tempfeatureversion";  oraCmd = new OracleCommand(sql, oraConn);  oraCmd.ExecuteNonQuery();  oraCmd.Dispose();  sql = @"insert into TEMPFEATUREVERSION  (select PRODFEATVER\_ID, PRODFEATVER\_ID, FOID, source\_repusage\_id ,PANELVER\_ID  from TMP\_PPR\_FEATURE\_VERSION\_INFO where STATE = 'Latest Version' and OBJECTCLASS = '$rncpanel')  ";  oraCmd = new OracleCommand(sql, oraConn);  oraCmd.ExecuteNonQuery();  oraCmd.Dispose(); |

Note the bind variable "panelversion\_id" in line 7 which is set in lines 9-12.  
Also note that the object class in line 24, in the sql string, is set to $rncpanel.  This can be changed to another feature class.

1. The next step is to get the geometry of the feature however the SDO\_GEOMETRY.SDO\_ORDINATES of SDO\_ORDINATES\_ARRAY must be mapped to a C# type via creation of a User Defined Type class and a User Defined Type Factory class.  Below are definitions of these two classes for an MDSYS.SDO\_ORDINATES\_ARRAY type:

|  |
| --- |
| /\*\*  \* Oracle User Defined Type class (UDT) for mdsys.sdo\_ordinate\_array type.  \* This class is based off of the UDT sample code provided with ODP.NET.  \* For more information about Oracle UDTs, see section "Oracle User-Defined Types"  \* in the Oracle Data Provider for .NET Developer's Guide  \*/  public class SDOOrdinateArray : IOracleCustomType, INullable  {  [OracleArrayMapping()]  public Double[] Array;  private OracleUdtStatus[] m\_statusArray;  public OracleUdtStatus[] StatusArray  {  get  {  return this.m\_statusArray;  }  set  {  this.m\_statusArray = value;  }  }  private bool m\_bIsNull;  public bool IsNull  {  get  {  return m\_bIsNull;  }  }  public static SDOOrdinateArray Null  {  get  {  SDOOrdinateArray obj = new SDOOrdinateArray();  obj.m\_bIsNull = true;  return obj;  }  }  public void ToCustomObject(OracleConnection con, IntPtr pUdt)  {  object objectStatusArray = null;  Array = (Double[])OracleUdt.GetValue(con, pUdt, 0, out objectStatusArray);  m\_statusArray = (OracleUdtStatus[])objectStatusArray;  }  public void FromCustomObject(OracleConnection con, IntPtr pUdt)  {  OracleUdt.SetValue(con, pUdt, 0, Array, m\_statusArray);  }  public override string ToString()  {  if (m\_bIsNull)  return "SDOOrdinateArray.Null";  else  {  string rtnstr = String.Empty;  if (m\_statusArray[0] == OracleUdtStatus.Null)  rtnstr = "NULL";  else  rtnstr = Array.GetValue(0).ToString();  for (int i = 1; i < m\_statusArray.Length; i++)  {  if (m\_statusArray[i] == OracleUdtStatus.Null)  rtnstr += "," + "NULL";  else  rtnstr += "," + Array.GetValue(i).ToString();  }  return "SDOOrdinateArray(" + rtnstr + ")";  }  }  }    /\*\*  \* Oracle User Defined Type Factory class (UDT) for mdsys.sdo\_ordinate\_array type.  \* This class is based off of the UDT sample code provided with ODP.NET.  \* For more information about Oracle UDTs, see section "Oracle User-Defined Types"  \* in the Oracle Data Provider for .NET Developer's Guide  \*/  [OracleCustomTypeMapping("MDSYS.SDO\_ORDINATE\_ARRAY")]  public class SDOOrdinateArrayFactory :  IOracleCustomTypeFactory, IOracleArrayTypeFactory  {  // IOracleCustomTypeFactory  public IOracleCustomType CreateObject()  {  return new SDOOrdinateArray();  }  // IOracleArrayTypeFactory Inteface  public Array CreateArray(int numElems)  {  return new Double[numElems];  }  public Array CreateStatusArray(int numElems)  {  // CreateStatusArray may return null if null status information  // is not required.  return new OracleUdtStatus[numElems];  }  } |

1. Get the geometry of the feature and load it into a List of Tuples.:

|  |
| --- |
| // Get varray of coordinates from the sdo\_geometry by calling p\_pprapi.GetGeometry  string sql = "SELECT t1.geom.sdo\_ordinates FROM TABLE(p\_pprapi.GetGeometry(:panelversion\_id)) t1";  OracleCommand oraCmd = new OracleCommand(sql, oraConn);  OracleParameter oraParam = new OracleParameter();  oraParam.OracleDbType = OracleDbType.Int32;  oraParam.Value = panelVersionId;  oraCmd.Parameters.Add(oraParam);  OracleDataReader oraReader = oraCmd.ExecuteReader();  List <Tuple<double, double>> chmrCoordinates = new List<Tuple<double, double>>();  if(oraReader.HasRows){  oraReader.Read();  // SDOOrdinateArray UDT for mdsys.sdo\_orginate\_array Oracle Data Type.  // See class definition further below  SDOOrdinateArray ordinates = (SDOOrdinateArray)oraReader.GetValue(0);  int i = 0;  while (i < ordinates.Array.Length)  {  if (i % 2 == 1)  {  chmrCoordinates.Add(new Tuple<double, double>(ordinates.Array[i - 1], ordinates.Array[i]));  }  i++;  }  } |

1. Retrieve the parameters needed for performing the transformation.  These parameters are:  panel ellipsoide name, x origin, y origin, x1 align, y1 align, x2 align , y2 align, source ellipsoid name, projection information, scale:

|  |
| --- |
| // First get parameters needed for transformation assistant API call.  string panelEllipsoidName;// PAPER\_PANEL.DATUM  double originX; // PAPER\_PANEL.ORIGIN\_X  double originY; // PAPER\_PANEL.ORIGIN\_Y  double alignX1; // PAPER\_PANEL.ALIGN\_LON1  double alignY1; // PAPER\_PANEL.ALIGN\_LAT1  double alignX2; // PAPER\_PANEL.ALIGN\_LON2  double alignY2; // PAPER\_PANEL.ALIGN\_LAT2  // Get panel ellipsoid name, origin, and alignment values  string sql = @"SELECT datum, origin\_x, origin\_y,  align\_lon1, align\_lat1, align\_lon2, align\_lat2  FROM paper\_panel  WHERE panelvr\_panelver\_id = :panelver\_id  ";  OracleCommand oraCmd = new OracleCommand(sql, oraConn);  OracleParameter oraParam = new OracleParameter();  oraParam.OracleDbType = OracleDbType.Int32;  oraParam.Value = panelVersionId;  oraCmd.Parameters.Add(oraParam);  OracleDataReader oraReader = oraCmd.ExecuteReader();  if (!oraReader.HasRows)  {  throw new Exception("Panel coordinate reference system information could not be found");  }  oraReader.Read();  panelEllipsoidName = (string)oraReader.GetValue(0);  originX = Decimal.ToDouble((decimal)oraReader.GetValue(1));  originY = Decimal.ToDouble((decimal)oraReader.GetValue(2));  alignX1 = Decimal.ToDouble((decimal)oraReader.GetValue(3));  alignY1 = Decimal.ToDouble((decimal)oraReader.GetValue(4));  alignX2 = Decimal.ToDouble((decimal)oraReader.GetValue(5));  alignY2 = Decimal.ToDouble((decimal)oraReader.GetValue(6));    // Get Source Ellipsoid datum from hpdcoordsys.  string sourceEllipsoidName; // HPDCOORDSYS.DATUM  sql = "SELECT datum FROM hpdcoordsys";  oraCmd = new OracleCommand(sql, oraConn);  oraReader = oraCmd.ExecuteReader();  if (!oraReader.HasRows)  {  throw new Exception("No Source Ellipsoid Found");  }  oraReader.Read();  sourceEllipsoidName = (string)oraReader.GetValue(0);    // Get projection Info with call to P\_PPRAPI.GETPANELVERSIONATTRIBUTES function  string projectionInfo; // P\_PPRAPI.GETPANELVERSIONATTRIBUTES  sql = @"select att\_value from table(p\_pprapi.GETPANELVERSIONATTRIBUTES(:panelversion\_id))  where acronym='PRJCTN'";  oraCmd = new OracleCommand(sql, oraConn);  oraParam = new OracleParameter();  oraParam.OracleDbType = OracleDbType.Int32;  oraParam.Value = panelVersionId;  oraCmd.Parameters.Add(oraParam);  oraReader = oraCmd.ExecuteReader();  if (!oraReader.HasRows)  {  throw new Exception("No projection info");  }  oraReader.Read();  projectionInfo = (string)oraReader.GetValue(0);  // Get panel scale with call to P\_PPRAPI.GETPANELVERSIONATTRIBUTES function  double scale; // P\_PPRAPI.GETPANELVERSIONATTRIBUTES  sql = @"select att\_value from table(p\_pprapi.GETPANELVERSIONATTRIBUTES(:panelversion\_id))  where acronym='PSCALE'";  oraCmd = new OracleCommand(sql, oraConn);  oraParam = new OracleParameter();  oraParam.OracleDbType = OracleDbType.Int32;  oraParam.Value = panelVersionId;  oraCmd.Parameters.Add(oraParam);  oraReader = oraCmd.ExecuteReader();  if (!oraReader.HasRows)  {  throw new Exception("No projection info");  }  oraReader.Read();  scale = Double.Parse((string)oraReader.GetValue(0));  oraParam.Dispose();  oraReader.Dispose();  oraCmd.Dispose(); |

1. Initialize the Transformation Assistant API:

|  |
| --- |
| int result;  // Initialise Transformation Assisten API passing the path to the installed datum.dat file  result = TransformationAssistantLib.TransformationAssistant.initializeEnvironment(  @"C:\Program Files\CARIS\HPD Suite\3.1\system\datum.dat");  if (result != 0)  {  throw new Exception("Unable to initialise Transformation Assistant API: " +  TransformationAssistantLib.TransformationAssistant.getErrorMessage(result));  } |

1. Transform each CHMR coordinate and load into new List of Typles:

|  |
| --- |
| // Call Transformation Assistant API to transform each chmr coordinate to an lldg coordinate  List<Tuple<double, double>> lldgCoordinates = new List<Tuple<double, double>>();  foreach (Tuple<double, double> chmrCoord in chmrCoordinates){  double lon = 0;  double lat = 0;  result = TransformationAssistantLib.TransformationAssistant.transformToProductLLDG(panelEllipsoidName,  sourceEllipsoidName, projectionInfo, scale, originX, originY, alignX1, alignY1,  alignX2, alignY2, chmrCoord.Item1, chmrCoord.Item2, ref lon, ref lat);  if (result != 0)  {  throw new Exception("Unable to transform coordinate: " +  TransformationAssistantLib.TransformationAssistant.getErrorMessage(result));  }  lldgCoordinates.Add(new Tuple<double, double>(lon, lat));  } |

# Sample C#.NET Project

The accompanying sample project is provided to help demonstrate a fully working program.  Extract the project files, open the TransformationAssistant.NET.sln solution and build the project.

The command line program that gets built is called TransformFeatureCoordinates.exe and takes the Oracle net service name, hpd username, password, and chart version id as parameters.

Example:

|  |
| --- |
| TransformFeatureCoordinates.exe orcl hpd hpd 2 Panel Version ID - Panel Name 6 - main Enter Panel Version ID: 6 CHMR Coordinates 1.0793163999997,10.7095568000002 7.76305630000024,15.1776947999999 12.1677751000002,7.86325729999999 5.34557040000027,3.03605359999983 1.07914699999992,10.5877704 LLDG Coordinates: Lon: -80.315323239537, Lat: 25.6662450424445 Lon: -80.1652205968557, Lat: 25.7571492711242 Lon: -80.0662999413485, Lat: 25.6083005338157 Lon: -80.2195122101836, Lat: 25.5099641772468 Lon: -80.3153270439022, Lat: 25.6637663167938 |

# Related articles

* Page:[Transform Panel Feature Coordinate With Transformation Assistant API](#scroll-bookmark-1)
* Page:[Oracle 12c - configuring extended string length](https://confluence.teledynecaris.com/display/~cpalmer/Oracle+12c+-+configuring+extended+string+length)