

Projecting Tool

Ein Werkzeug zur Koordinatentransformation für Coverages
vom Gauß-Krüger- nach UTM -Koordinatensystem

Ines Imkamp und Gudrun Künne
GICON – Großmann Ingenieur Consult GmbH
Niederlassung Leipzig

Leipzig, am 10. Juni 2010

- Präsentationsschwerpunkte
 - GICON – Großmann Ingenieur Consult GmbH
 - Niederlassung Leipzig der GICON
 - Ausgangssituation
 - Vergleich verschiedener Umsetzungsmöglichkeiten
 - Warum .NET-Anwendung
 - Erläuterung der Umsetzung
 - Live-Vorführung
 - Ausblick
 - Diskussion

- Unabhängiges **Consulting- und Engineeringunternehmen mit Sitz in Dresden**
- Niederlassungen und weitere verbundene Unternehmen in ganz Deutschland



- ca. 150 Mitarbeiter
- Geschäftsbereiche
 - Anlagen- und Genehmigungsplanung
 - Energie und Umwelt
 - Boden- und Gewässermanagement
 - Erkundung und Monitoring
 - Geotechnik und Baumanagement
 - Technische Informatik
 - Fachinformationssysteme (NL Leipzig)

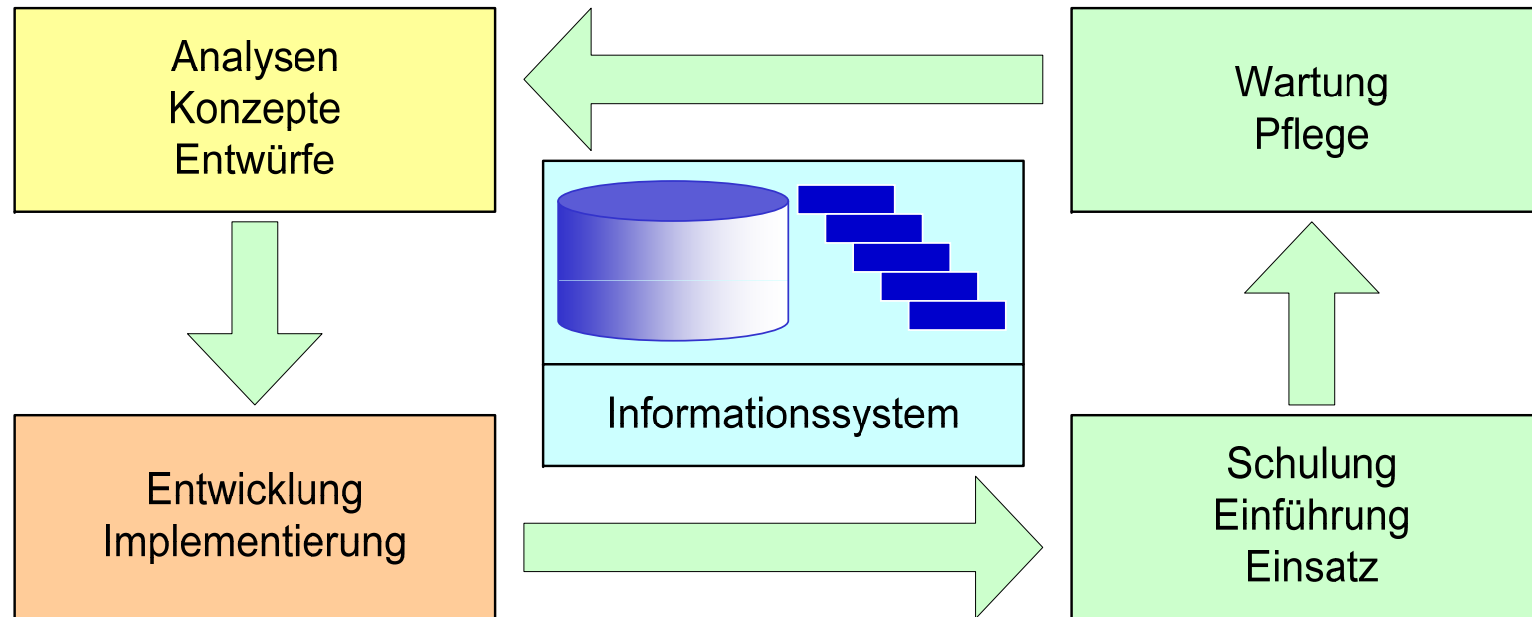


GICON Großmann Ingenieur Consult GmbH
Niederlassung Leipzig
Berliner Straße 81 a
(seit Juni 2008)

NOELL Umweltdienste GmbH
PREUSSAG AG
BISANTECH

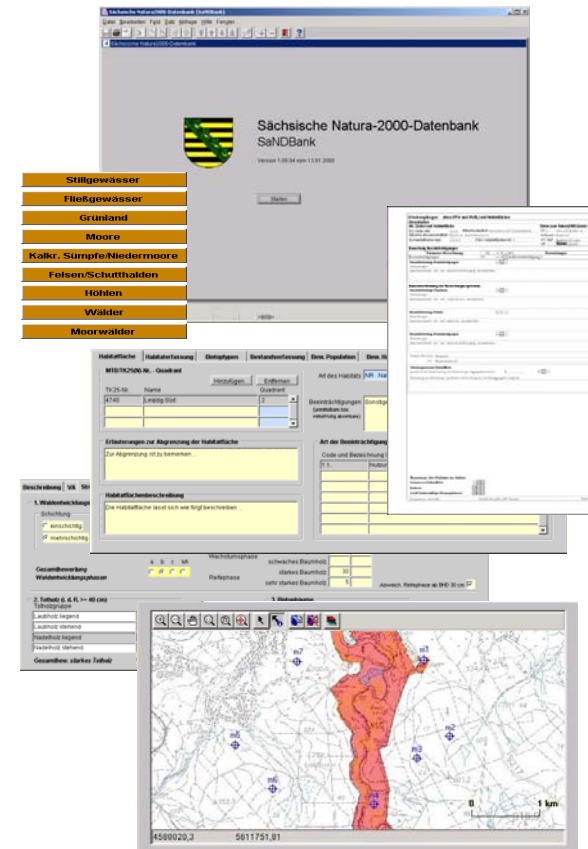
Leipzig, Berliner Str. 81a



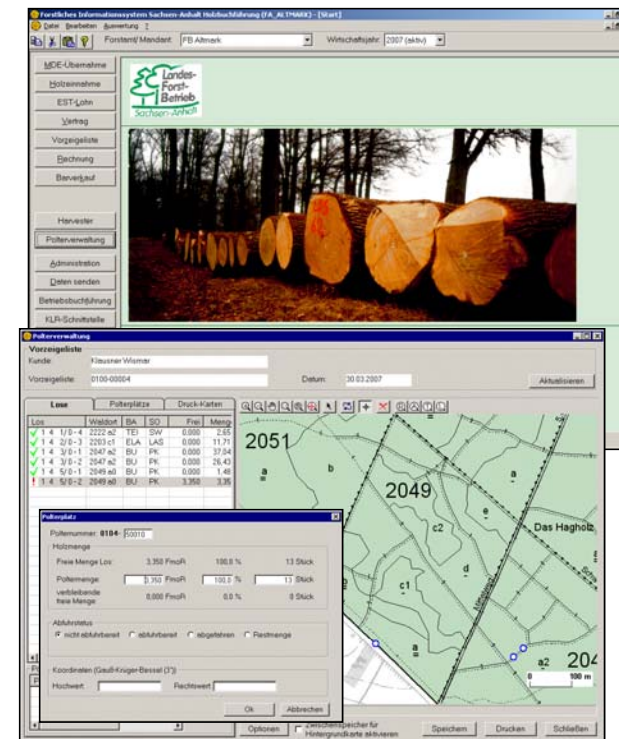


- Datenbanken
- Softwareentwicklung
- Geografische Informationssysteme (GIS)
- Microsoft, Oracle, ESRI ArcGIS, u.a.

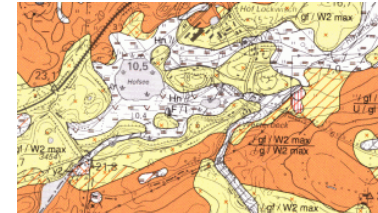
- Altlasten
- Bergbau
- Boden, Rohstoffe
- Forstwirtschaft, Landwirtschaft
- Klimafolgen und Anpassung
- Luft- und Lärmbelastung
- Naturschutz/ NATURA 2000
- Umweltradioaktivität
- Umweltschutz
- Wasserwirtschaft, Gewässerökologie



- **Sächsisches Oberbergamt**
Bergbauinformationssystem SBIS
mit Nachnutzung im LBGR Brandenburg und LGB Rheinland-Pfalz
- **Landesforstbetrieb Sachsen-Anhalt**
Forstliches Informationssystem
- **LfULG Sachsen**
FIS Boden, FIS Rohstoffe, KANARAS,
NATURA 2000
- **UBG und LTV Sachsen**
BIO-DB und LimnoBase
- **Umweltbundesamt**
FISKA und BEU-MESAP
- **UIS und Standortinformationssysteme**
für chemische Industrie



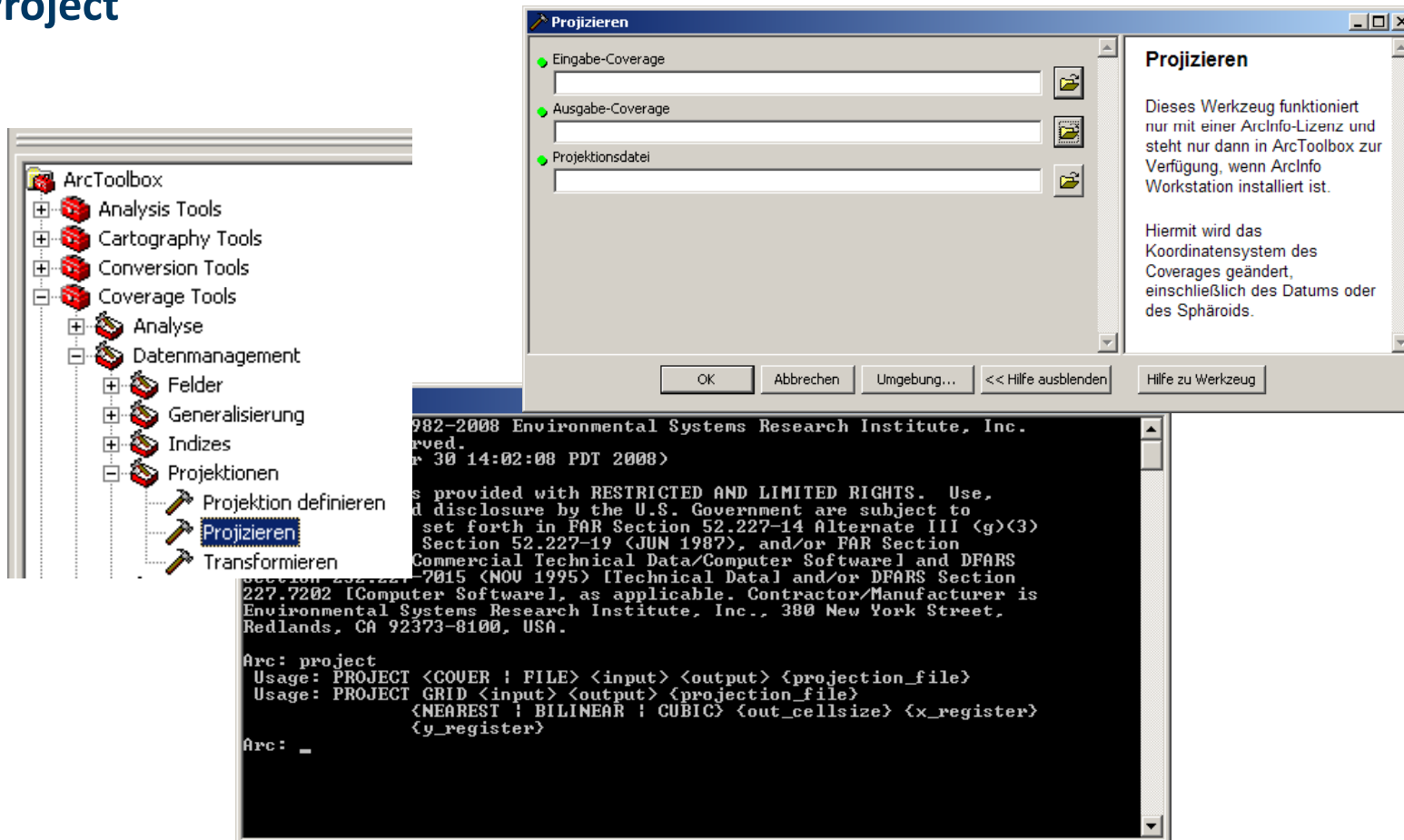
- Bodeninformationssystem Mecklenburg-Vorpommern (BISMV) für das Landesamt für Umwelt, Naturschutz und Geologie Mecklenburg-Vorpommern (LUNG)
 - AML-Anwendung zur Erzeugung und Druck von Geologischen Kartenblättern (GK25, GK50, KOR50 etc.)
 - Übernahme Lösung Niedersachsen (NIBIS), Anpassung und Wartung des Systems für das LUNG durch GICON → BISMV
 - Anwendung arbeitet mit dem Datenformat Coverages
 - Umstellung auf ETRS89 war gefordert → Wartungsleistung



- Einzel per Hand
- Model Builder
- .NET + ArcObjects – Programm bzw. -Erweiterung

Vergleich verschiedener Umsetzungsmöglichkeiten – einzeln per Hand

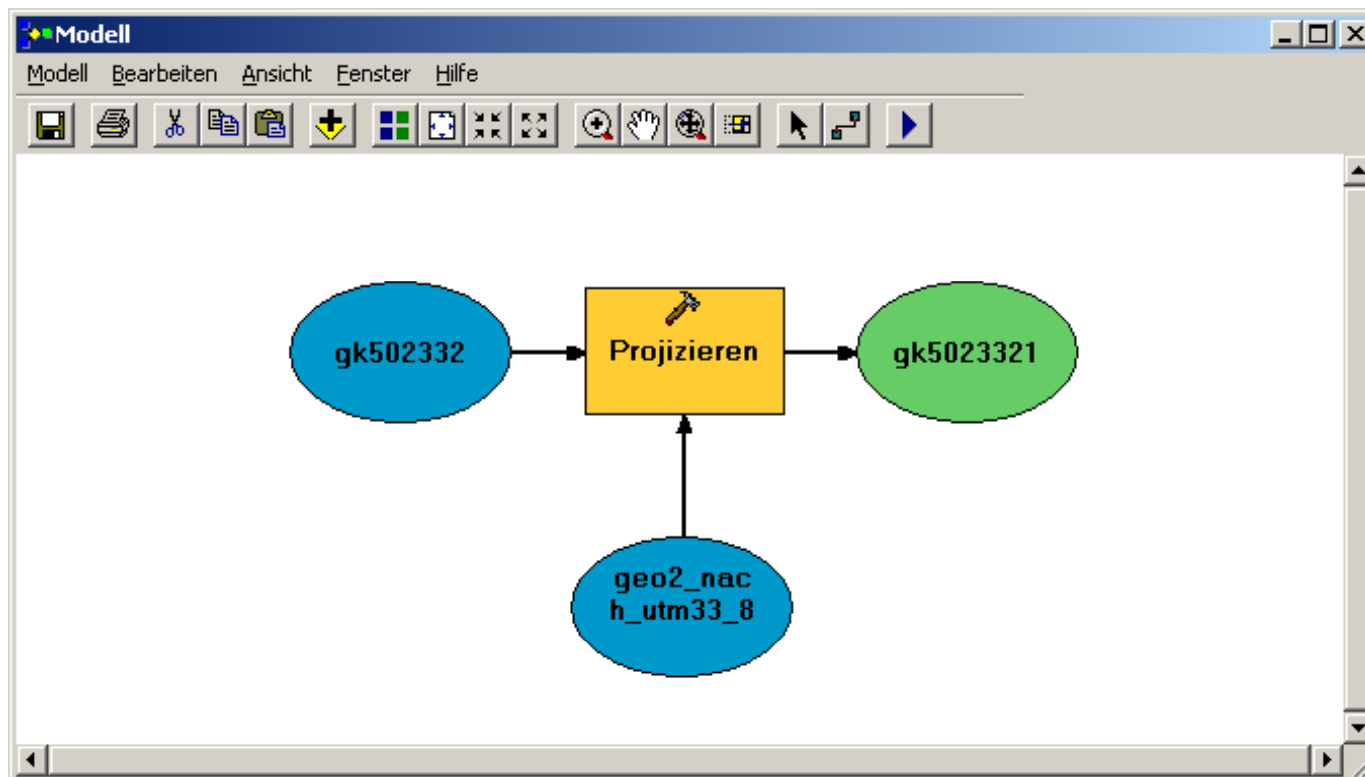
- Anwendung unter ArcToolBox oder ArcInfo Workstation → Funktion **Project**



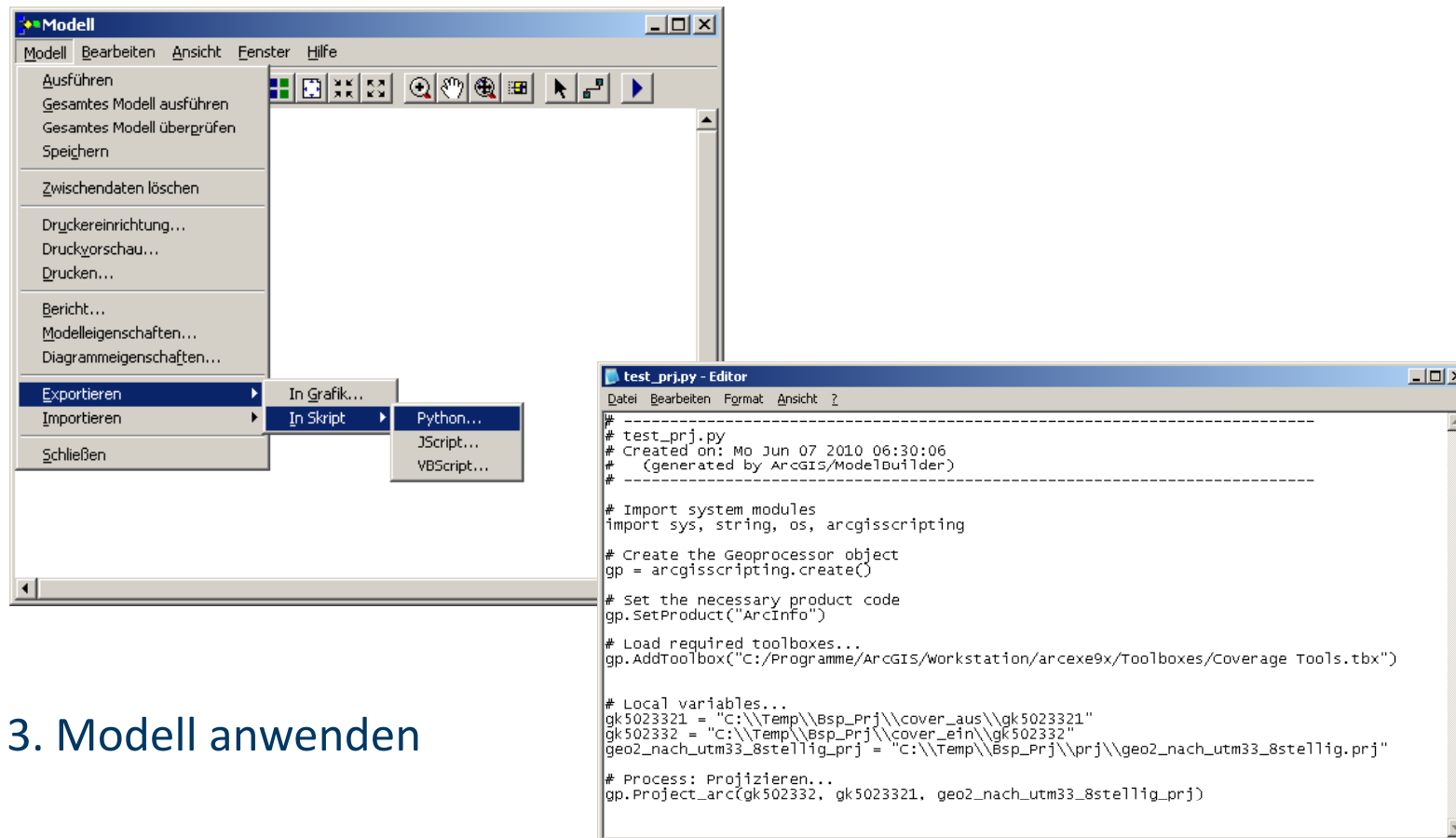
- Aufbau Projektionsdatei für Coverages bsp.prj

```
input
projection transverse
units meters
xshift -4500000
spheroid bessel
parameters
1
12 00 00
0 00 00
0
0
output
projection geographic
units dd
spheroid bessel
parameters
end
```

1. Modell erzeugen



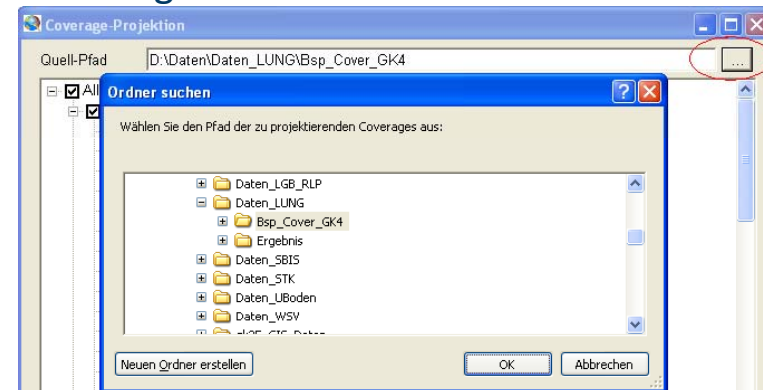
2. Modell exportieren



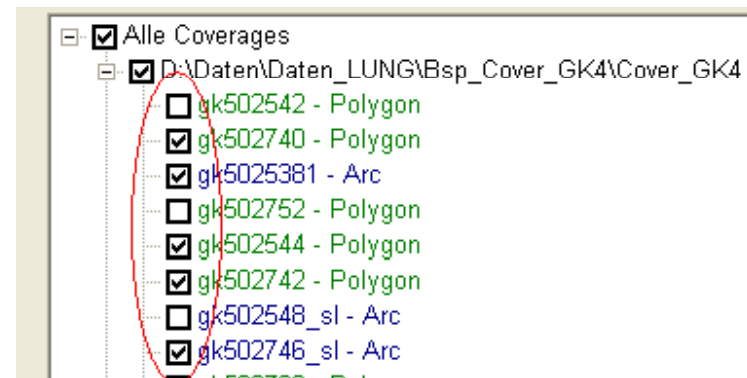
3. Modell anwenden

Warum Microsoft .NET Framework, C# ?

- Komfortable Programmierumgebung für Dialoge
 - flexible Auswahl der Quell- und Ziel-Verzeichnisse möglich
(Standardeinstellungen werden in der Windows-Registry abgelegt)



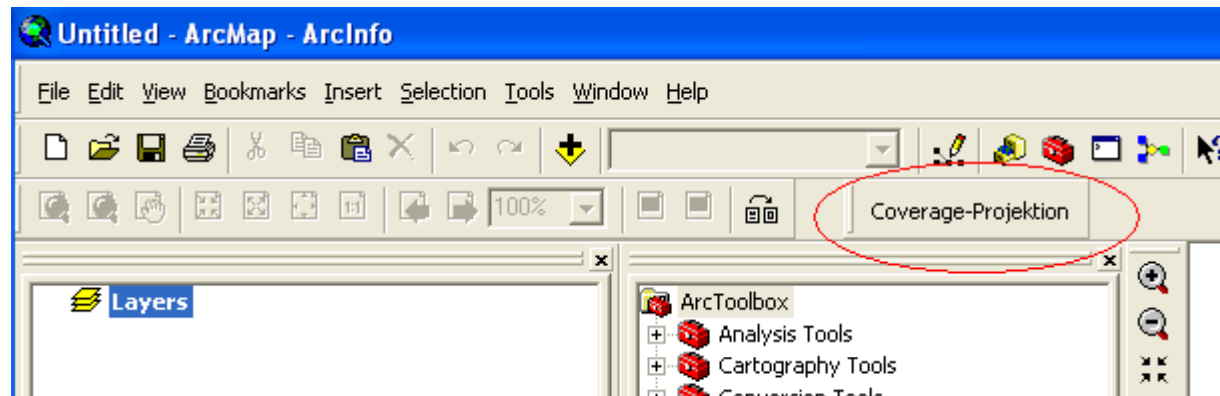
- Auswahl der Coverages aus verschiedenen Unter-Verzeichnissen möglich
- flexible Auswahl der zu transformierenden Coverages durch Mausklick



- Mit Hilfe des Geoprocessors können mehrere Tools auf mehrere Coverages nacheinander angewendet werden.

Warum Erweiterung zu ArcMap ?

- Stand-Alone-Anwendung
 - muss über das Windows-Startmenü extra gestartet werden
 - muss die benötigte ArcGIS Lizenz prüfen
- Erweiterung zu ArcMap
 - In ArcMap wird die registrierte neue Toolbar leicht gefunden.

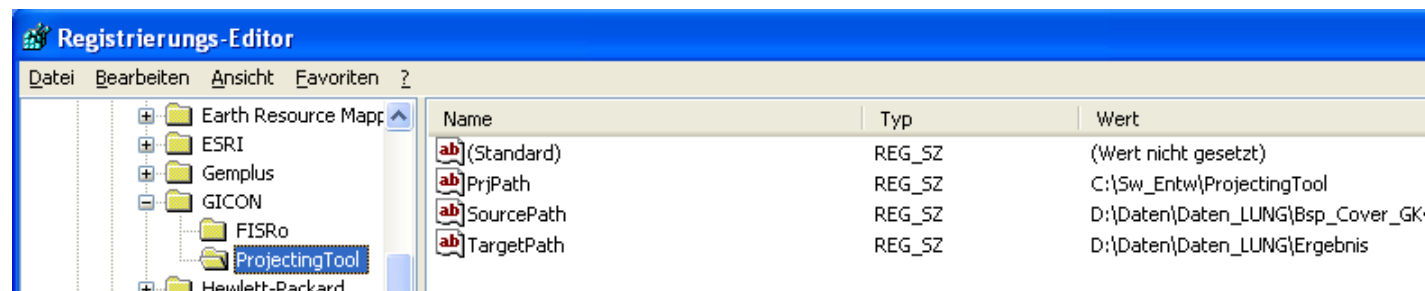


- Vorhandensein der ArgGIS-Lizenz ist abgesichert

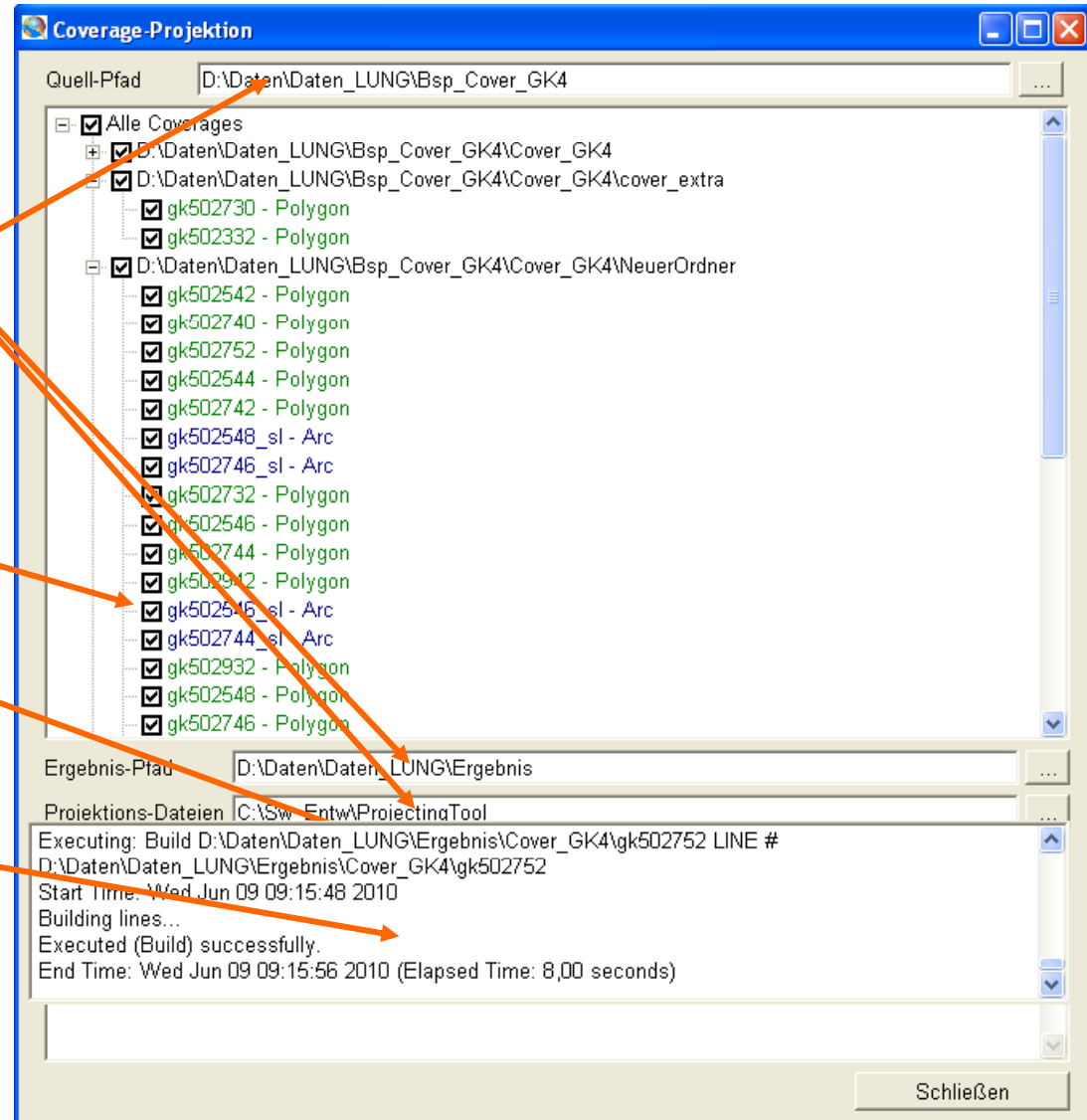
- Auslieferung der Anwendung in Form einer Datei „ProjectingTool.dll“ und der Hilfsdatei „regProjectingTool.bat“
 - Mit Doppelklick der **regProjectingTool.bat** wird die **ProjectingTool.dll** registriert.
 - In ArcMap wird eine neue Toolbar „ProjectingTool“ mit einer Schaltfläche „Coverage-Projektion“ bereitgestellt.



- Eintrag der Standard-Pfade in der Windows Registry
 - Die Standardwerte der Pfade für Quelle, Ergebnis und Projektionsdateien werden in der Windows Registry eingetragen.



- Klick auf die Schaltfläche
 - Anzeige des Dialogs
 - Setzen der Voreinstellungen aus der Registry
 - Alle Coverages des aktuellen Verzeichnisses und aller Unterverzeichnisse werden gelesen
 - Start der Anwendung
 - Es wird ein detaillierter Bericht erstellt und gleichzeitig im Dialog angezeigt



■ Geoprocessing = Geoverarbeitung

- Geoprocessor ist ein Bestandteil von ArcMap, ArcCatalog, ArcScene, ArcGlobe
- Ermöglicht die Definition von Arbeitsabläufen von sich wiederholenden GIS-Aufgaben wie
 - Datenkonvertierungen
 - Projektionen
 - Räumliche Überlagerungen (Verschneidung) usw.
- Die Ausführung erfolgt mit Hilfe von **Tools** aus **Toolboxen** in unserem Fall aus der Toolbox „**Coverage Tools.tbx**“
- Die Namen der **Tools** einer **Toolbox** kann man mit der Funktion **ListTools()** des Geoprocessors ermitteln.
Folgende Tools aus der Toolbox „**Coverage Tools.tbx**“ werden in der vorgestellten Anwendung verwendet:
 - DefineProjection_management
 - Project_arc (mit 3 verschiedenen Projektionsdateien)
 - Build_arc
- Es wird ...
 1. eine Liste der erforderlichen Parameter (Eingabe-Coverage, Ausgabe-Coverage, Projektionsdatei) erstellt,
 2. das Tool mit der Funktion Execute() des Geoprocessors ausgeführt.

- Geoprocessor initialisieren

```
// Geoprocessor initialisieren
Geoprocessor oGP = new Geoprocessor();
oGP.AddToolbox(sTbxPath);          // z.B.: C:\Programme\ArcGIS\arcexe9x\Toolboxes\Coverage Tools.tbx
oGP.OverwriteOutput = true;

// Deklaration eines Vararray zur Übergabe der Parameter
IVariantArray pVarArray;
int nCount = 1;
```

- durch die Liste der ausgewählten Coverages

```
// durch die Liste der Coverages
foreach (string[] sCoverInfo in lstCovers)
{
    // string-Array enthält: 0 = Quell-Pfad, 1 = Ziel-Pfad, 2 = Cover-Name, 3 = Geometrietyp
    sInputPath = sCoverInfo[0];
    sOutputPath = sCoverInfo[1];
    sGeomTyp = sCoverInfo[3];
    sInputCover = sInputPath + @"\ " + sCoverInfo[2];
    sOutputCover = sOutputPath + @"\ " + sCoverInfo[2];

    // Falls Ausgabe-Directory nicht existiert, erstellen
    if (!Directory.Exists(sOutputPath))
        Directory.CreateDirectory(sOutputPath);

    // MouseCursor auf "Wait" setzen
    IMouseCursor pMouseCursor = new MouseCursorClass();
    pMouseCursor.SetCursor(2);

    // Startzeile für neues Coverage in Berichts-Datei und Dialog
    sMessage = Environment.NewLine + "***** (" + nCount.ToString() + ") ***** Coverage: " + sCoverInfo[2]
        + "*****" + Environment.NewLine + " von " + sInputPath + " nach " + sOutputPath + Environment.NewLine;
    GISTools.WriteTextFile(sMessageFile, sTempPath, sMessage, ref txtMsg, false);
    nCount++;
}
```

- Das Tool *DefineProjection_management* ermöglicht das Zuweisen einer Projektion zu einem Coverage.
 - Parameter:
 - das zu bearbeitende **Coverage**
 - die **Projektionsdatei „GK4.PRJ“**
 - Am Ende werden die Meldungen des Geoprocessors in die Berichtsdatei geschrieben.
(die Funktion *GISTools.WriteTextFile()* hängt die übergebene Message an die Berichtsdatei und an die Meldungsbox im Dialog an)

```
sMessage = Environment.NewLine + "----- DefineProjection_management --|-----"  
// Parameter in VarArray schreiben ...  
pVarArray = new VarArrayClass();  
pVarArray.Add(sInputCover);  
pVarArray.Add(sPrj01);  
pMouseCursor.SetCursor(2);  
// ... und den Geoprocessor mit dem Tool und den Parametern ausführen  
oGP.Execute("DefineProjection_management", pVarArray, null);  
// Meldungen des Geoprocessors in Datei und TextBox schreiben  
if (oGP != null && oGP.MessageCount > 0)  
    for (int i = 0; i < oGP.MessageCount; i++)  
        sMessage += Environment.NewLine + oGP.GetMessage(i);  
GISTools.WriteTextFile(sMessageFile, sTempPath, sMessage, ref txtMsg, false);
```

- Das Tool **Project** wird mit 3 verschiedenen Projektionsdateien ausgeführt.
Ausführung mit der ersten Projektionsdatei
 - Parameter:
 - das zu bearbeitende **Coverage**
 - ein temporäres **Ergebnis-Coverage**
 - die **Projektionsdatei „4_nach_geo1.prj“**
 - zur Transformation vom Gauß-Krüger-Koordinatensystem, 4. Meridianstreifen in ein geographisches Koordinatensystem
 - Am Ende werden die Meldungen des Geoprocessors in die Berichtsdatei geschrieben.

```
sMessage = Environment.NewLine + "----- Project_arc -----"  
// Parameter in VarArray schreiben ...  
pVarArray = new VarArrayClass();  
pVarArray.Add(sInputCover);  
pVarArray.Add(sTempCover1);  
pVarArray.Add(sPrj02);  
pMouseCursor.SetCursor(2);  
// ... und den Geoprocessor mit dem Tool und den Parametern ausführen  
oGP.Execute("Project_arc", pVarArray, null);  
// Meldungen des Geoprocessors in Datei und TextBox schreiben  
if (oGP != null && oGP.MessageCount > 0)  
    for (int i = 0; i < oGP.MessageCount; i++)  
        sMessage += Environment.NewLine + oGP.GetMessage(i);  
GISTools.WriteTextFile(sMessageFile, sTempPath, sMessage, ref txtMsg, false);
```

- Das Tool **Project** wird mit der zweiten Projektionsdatei ausgeführt
 - Parameter:
 - das temporäre **Ergebnis-Coverage** der ersten Projektion
 - ein zweites temporäres **Ergebnis-Coverage**
 - die **Projektionsdatei „geo1_nach_geo2.prj“**
 - zur Transformation von einem geographischen Koordinatensystem in ein anderes geographisches Koordinatensystem mit Wechsel des Sphäroiden und des Datums
 - Am Ende werden die Meldungen des Geoprocessors in die Berichtsdatei geschrieben.

```
sMessage = Environment.NewLine + "----- Project_arc -----"  
// Parameter in VarArray schreiben ...  
pVarArray = new VarArrayClass();  
pVarArray.Add(sTempCover1);  
pVarArray.Add(sTempCover2);  
pVarArray.Add(sPrj03);  
pMouseCursor.SetCursor(2);  
// ... und den Geoprocessor mit dem Tool und den Parametern ausführen  
oGP.Execute("Project_arc", pVarArray, null);  
// Meldungen des Geoprocessors in Datei und TextBox schreiben  
if (oGP != null && oGP.MessageCount > 0)  
    for (int i = 0; i < oGP.MessageCount; i++)  
        sMessage += Environment.NewLine + oGP.GetMessage(i);  
GISTools.WriteTextFile(sMessageFile, sTempPath, sMessage, ref txtMsg, false);
```

- Das Tool **Project** wird mit der dritten Projektionsdatei ausgeführt
 - Parameter:
 - das temporäre **Ergebnis-Coverage** der zweiten Projektion
 - das **Ergebnis-Coverage**
 - die **Projektionsdatei „geo2_nach_utm33.prj“**
 - zur Transformation vom geographischen Koordinatensystem in nach UTM33
 - Am Ende werden die Meldungen des Geoprocessors in die Berichtsdatei geschrieben.

```
sMessage = Environment.NewLine + "----- Project_arc -----"  
// Parameter in VarArray schreiben ...  
pVarArray = new VarArrayClass();  
pVarArray.Add(sTempCover2);  
pVarArray.Add(sOutputCover);  
pVarArray.Add(sPrj04);  
pMouseCursor.SetCursor(2);  
// ... und den Geoprocessor mit dem Tool und den Parametern ausführen  
oGP.Execute("Project_arc", pVarArray, null);  
// Meldungen des Geoprocessors in Datei und TextBox schreiben  
if (oGP != null && oGP.MessageCount > 0)  
    for (int i = 0; i < oGP.MessageCount; i++)  
        sMessage += Environment.NewLine + oGP.GetMessage(i);  
GISTools.WriteTextFile(sMessageFile, sTempPath, sMessage, ref txtMsg, false);
```

- Wenn der Geometrietyt des Coverages Polygon ist, muss die Topologie wieder aufgebaut werden
 - Parameter:
 - **LINE** der zweiten Projektion
 - das **Ergebnis-Coverage**
 - Parameter:
 - **LINE** der zweiten Projektion
 - das **Ergebnis-Coverage**

```
if (sGeomTyp == "Polygon") // wenn Geometrietyt Polygon, muss 2 mal Build ausgeführt werden
{
    // Build Arc
    sMessage = Environment.NewLine + "----- Build_arc (LINE) -----"
    pVarArray = new VarArrayClass();
    pVarArray.Add(sOutputCover);
    pVarArray.Add("LINE");
    pMouseCursor.SetCursor(2);
    oGP.Execute("Build_arc", pVarArray, null);
    if (oGP != null && oGP.MessageCount > 0)
        for (int i = 0; i < oGP.MessageCount; i++)
            sMessage += Environment.NewLine + oGP.GetMessage(i);
    GISTools.WriteTextFile(sMessageFile, sTempPath, sMessage, ref txtMsg, false);

    // Build Poly
    sMessage = Environment.NewLine + "----- Build_arc (POLY) -----"
    pVarArray = new VarArrayClass();
    pVarArray.Add(sOutputCover);
    pVarArray.Add("POLY");
    pMouseCursor.SetCursor(2);
    oGP.Execute("Build_arc", pVarArray, null);
    if (oGP != null && oGP.MessageCount > 0)
        for (int i = 0; i < oGP.MessageCount; i++)
            sMessage += Environment.NewLine + oGP.GetMessage(i);
    GISTools.WriteTextFile(sMessageFile, sTempPath, sMessage, ref txtMsg, false);
}
```

- Ist anpassbar für andere Koordinatensysteme
- Leider können nur maximal 17 Coverages mit einem Mal transformiert werden. Dann erscheint folgende Fehlermeldung



und die Anwendung muss erneut gestartet werden.

Aus den Diskussionen auf <http://forums.esri.com/> lässt sich schließen, dass auch andere Nutzer von diesem Problem betroffen sind und noch nach einer Lösung suchen.

Vielen Dank für Ihre Aufmerksamkeit

GICON Großmann Ingenieur Consult GmbH
Niederlassung Leipzig
Berliner Straße 81 a
04129 Leipzig

Tel. (0341) 90 999 0

i.imkamp@gicon.de

g.kuenne@gicon.de