Dynamic User Interface Creation with ArcObjects Based on Database Constraints

Joerg Sparenborg

Abstract

This paper describes a new framework for creating generic graphical user interfaces with ArcObjects and VBA for database access of spatial and non-spatial data recorded in a Geodatabase. The user interface elements are defined only by the database attributes and are created dynamically at runtime. Depending on field data types and the SDE specific features, such as Subtypes, Domains or Relationship Classes, appropriate VBA controls are selected automatically. Additionally, user input is validated based on consistency constraints stored in the database as corresponding field descriptions. This framework has been implemented for an operational multiuser environment for planning environmental reconnaissance flights and managing aerial photos.

Introduction

The tool described in this paper has been developed within the context of a research cooperation between the Swiss Federal Office of Topography and the Swiss Federal Research Institute WSL in the project LUBIS (LUftBild-InformationsSystem). LUBIS is a Geographic Information System (GIS) for recording and managing metadata of aerial photograph flights as well as the aerial photos taken thereby. The entire workflow of flight planning, archiving of resulting data and lending of aerial photos is supported by the system. The application consists of the following modules:

- **Flight planning:**
  This module contains methods for planning of aerial photograph flights. The planned flight data is used as an annotation file for the aerial photo labelling. Therefore the data is exported from LUBIS and read into the camera's guidance system on the aircraft.

- **Archiving:**
  After the flight the recorded metadata is imported into LUBIS.

- **Interface modules:**
  Different interfaces enable data exchange between LUBIS and the flight navigation and camera system on the aircrafts. To enable international data exchange coordinates of aerial photos taken outside Switzerland are transformed into WGS 84 format (World Geodetic System of 1984). An additional interface enables data export to Microsoft Excel.

- **Air traffic control module:**
  Air traffic control maps are automatically generated for different aircraft types and air control zones. The maps can be sent as jpeg-files to the air traffic control.

- **Aerial photo library module:**
  The library module handles lending of aerial photos. All operational sequences are integrated such as extension of the term, photo return, reminder list generation, etc.

- **Query tools:**
  Different query tools offer simple access to the data. Apart from spatial queries either specifying coordinates or selecting by dragging a box on the screen more complex queries may be composed by a particular query tool. With the assistance of a map tool the results can be printed. The example below (see figure 1) shows the GUI of the application LUBIS. Aerial photograph flights represented as lines and locations of belonging aerial photos represented as symbols are displayed as query result by a dragging box selection. Several information tools identify features by pointing to them. The user gets a list with attributes and appropriate values of identified features.

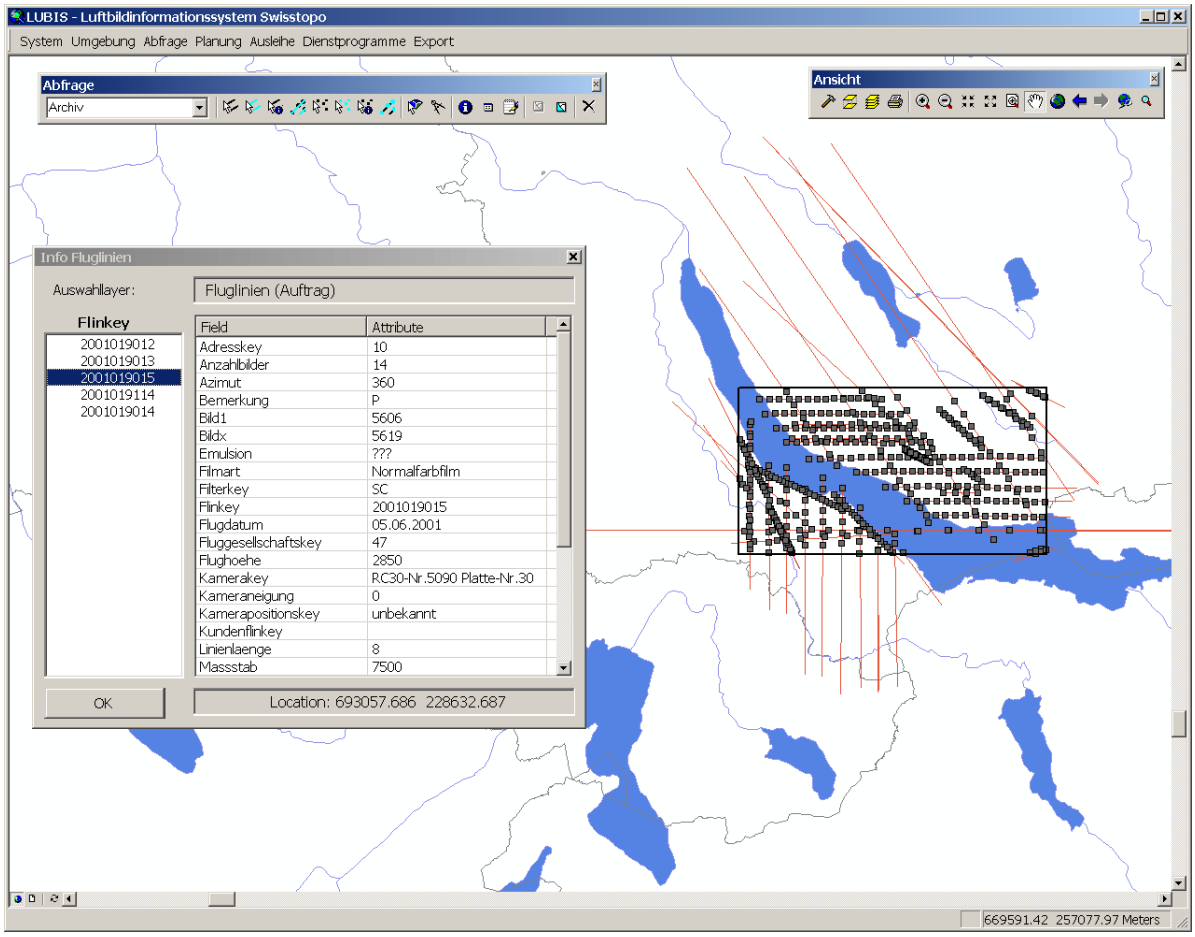


Figure1: Display of aerial photograph flights and locations of aerial photos as a query result.

- **Utilities:**
  Different tools enable duplication of planned flights, handling of planned but not realized flights and subsequent editing of flights already planned.

- **Database tool:**
  This tool handles direct access to the database. It is described in detail below.

LUBIS was designed and implemented as a multi-user information system. All users must be able to access the data concurrently. The system is realized as a client / server architecture using ArcGIS 8.3 as client on PC. ESRI Geodatabase is used as the data model. All data (spatial and non-spatial) are stored in an Oracle database. Data access and versioning are managed by ArcSDE.
The application has been developed completely in VisualBasic for Applications (VBA) and consists at present of 51 forms, 28 modules and 3 class modules with approximately 57000 lines of code. We extended ArcGIS with our own commands, tools and menus. The VBA-code is embedded within a mxd-project file.
LUBIS is in operation at the Federal Office of Topography since beginning of 2004. At present there are about 10000 flight lines and 150000 datasets of aerial photos stored in the database. About 5000 new datasets are created annually.

Database tool

The database tool enables direct access to the database. It is used for getting an overview about existing foreign keys and domain values on the one hand and for editing existing datasets and creating new records in the attribute tables on the other hand. The data base integrity has to be ensured at any time.
There are three types of users working with LUBIS: ArcSDE and database administrators, experienced users and standard users. This tool is only intended for experienced users, who can make changes in the database without assistance of an administrator. Administrators, in contrast, use standard tools such as ArcCatalog in order to edit the database. The standard user does not have an access to this tool because he is not allowed to change anything in the database.

Design-Requirements

The tool must fulfil the following requirements:

- Authentication: Only specified users have access.
- ObjectIDs are not permitted for editing.
- Exclusion list of attributes. Edits are only allowed for a restricted set of attributes.
- Primary keys must be created automatically and are not allowed to be edited.
- Input of foreign keys is implemented by ComboBoxes in order to ensure data integrity.
- Deletion of datasets is only permitted in feature classes.
- Input of domain values is managed by ComboBoxes.
- Export of attribute tables to Microsoft Excel.

The integrated table tool of ArcMap doesn't fulfil all of these requirements. Nonexistent foreign key values can be inserted into the database without any problems. Deletion of datasets can't be prevented. Furthermore, access to particular attributes can't be blocked.
For these reasons we developed a new database tool.
Forms are usually created in VBA with the GUI builder. During software development GUI elements (Buttons, Labels, ComboBoxes, TextBoxes, etc.) are interactively placed by drag and drop. VBA code is created subsequently and linked with the GUI elements. At runtime the visibility or state (e.g. locked, enabled) of GUI elements is changed dynamically according to the user input and business logic. If we had created for each feature class, attribute table and domain a form with associated code the application mxd-file would have increased accordingly. In addition, during software development it was already clear that several attribute tables and feature classes had to be extended by new attributes. This would have implicated extensive adjustments in the GUI as well as in the code.

Concept

On the basis of these requirements and conditions we had the idea to develop a universal database tool in which the GUI is created dynamically at runtime. Starting point is a table containing a number of attributes. In a first step the composition of the table is analyzed. Domains, attribute names, length, precision and data type are identified for each attribute. Table links are identified by analyzing relationship classes. Subsequently, an appropriate GUI element is created. As GUI elements ComboBoxes for lists of different values such as domain values or foreign key values, TextBoxes and Labels are used. The result of the algorithm is a form which contains GUI elements for each attribute.
A key requirement for successful operation is a correctly modelled database. Feature classes and attribute tables must be linked by relationship classes.
The next section describes the algorithm, followed by the description of input verification and dataset creation.

Algorithm for GUI generation

The complete algorithm is fully documented in the appendix as a flow chart. In the following we describe the underlying algorithm to create a GUI at runtime, which is exemplified by means of access of the attribute table *film*. The procedure with feature classes is identical up to small deviations. Most of the steps of the algorithm are documented with small code excerpts listed after the description of the program step.

1. Definition of an empty form
   Starting point is an empty form (see figure 2) on which only a save- and a cancel-button are predefined. Methods are implemented for both buttons managing mouse click events. The method called by clicking the save-button is described in section *Input check* below. This form is embedded in the mxd-file.

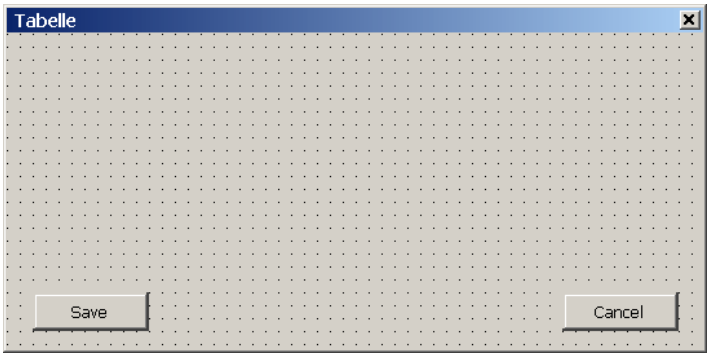


Figure 2: Empty form

2. Identify the primary key
   In the first place the primary key of the underlying attribute table is identified. Therefore all relationship classes are scanned for the current table name. If an appropriate relationship class is found the attribute name is recorded as primary key for further processing (see section 4.4). Otherwise there does not exist any primary key besides the Objectid.

3. Identify the foreign keys

In the next step a list with all foreign key names linked with this attribute table is generated by again scanning the relationship classes. The code excerpt below shows the analysis of enumeration of relationship classes linked with the table. Identified foreign key names are recorded in a list named `listFKeys`. We implemented this list as a new data type `list` which is realised as a dynamic list.

```
Set pObjectClass = pTable
Set pEnumRC = pObjectClass.RelationshipClasses(esriRelRoleDestination)
Set pRelClass = pEnumRC.Next
Do While Not pRelClass Is Nothing
  strDestTable = pRelClass.DestinationClass.AliasName
  If strDestTable = pObjectClass.AliasName Then
        listFKeys.AddItem pRelClass.OriginForeignKey
  End If
  Set pRelClass = pEnumRC.Next
Loop
```

4. Attribute handling

The following work steps are accomplished in a do-while loop for each attribute in the table.

4.1 Domain

If the attribute contains values of a domain, a list with all existing values of that domain is generated by a call to the method GetCodedValueDomainList(). We implemented this method for getting a list of all domain values. Subsequently, a ComboBox is created as a GUI element consisting of the value list created by the method call. This procedure guarantees that the list entries of the ComboBox only consist of assigned domain values. Currently, only the CodedValueDomain is implemented, since other domains are not used in our data model. The code excerpt below shows the generation of a list of domain values, the creation of a GUI element and the setting of its properties.

```
Set pDomain = pField.Domain
strDName = pDomain.Name
Set listDomainValues = modDomain.GetCodedValueDomainList(strDName)
Set pControl = Controls.Add("Forms.ComboBox.1", strControl, Visible)
With pControl
  Set .Font = pFont
  .Left = intXPos
  .Top = intYPos
  .Height = 16
  .Width = 120
  .Visible = True
  .MatchEntry = fmMatchEntryComplete
  .MatchRequired = True
  .List = listDomainValues.GetAllItems
End With
```

4.2 Foreign key

In the next step it is checked if the attribute contains foreign keys from another attribute table. In addition, the attribute table name is compared with the list of the foreign keys created in step 3. If the attribute contains a foreign key, a list with all key values is generated. Subsequently, a ComboBox is created as a GUI element consisting of the value list created.

This procedure guarantees that the list entries of the ComboBox only consist of assigned key values. Additionally, the properties of the ComboBox are defined, so that only assigned values can be selected.

4.3 ObjectID

If the attribute contains the Objectid, a TextBox is created as a GUI element. The properties of this TextBox are defined such that the value can't be edited by the user.

```
Set pControl = Controls.Add("Forms.TextBox.1", strControl, Visible)
If pField.Type = esriFieldTypeOID Then
   With pControl
         .Locked = True
   End With
End If
```

4.4 Primary key

If the current attribute name is identified as the primary key by comparing with the value recorded in step 2, then a TextBox is created as a GUI element. Subsequently a new key is generated by a call to the method `GetNextKey()` and filled in the TextBox. The user is prevented from editing this value. We implemented this method to automatically generate new keys in an attribute table.

```
Set pControl = Controls.Add("Forms.TextBox.1", strControl, Visible)
If pField.Name = strPKey Then
   intNewKey = modTable.GetNextKey(pDataSet.Name, pField.Name)
   With pControl
         .Locked = True
         .Text = intNewKey
   End With
End If
```

4.5 Locked attributes

The tool provides the opportunity of recording names of non-editable attributes in a list. The current attribute is checked for being included in this particular list. If it is in the list, the TextBox is locked and the user can not edit the value.

4.6 Other attributes

In all remaining cases a TextBox is created as a GUI element.

```
Set pControl = Controls.Add("Forms.TextBox.1", strControl, Visible)
If pField.Type = esriFieldTypeString Then
   pControl.MaxLength = pField.Length
Else
   pControl.MaxLength = pField.Precision
End If
```

4.7 Create Label

A Label is created for the GUI element, containing the attribute name.

```
Set pLabel = Controls.Add("Forms.Label.1", "label", Visible)
pLabel.Caption = pField.Name
```

5 Size of form

Finally, the form size is computed according to the created GUI elements and adapted accordingly. The save- and cancel-button are shifted to the lower edge of the form.

The completed form of the attribute table *film* has the following appearance:

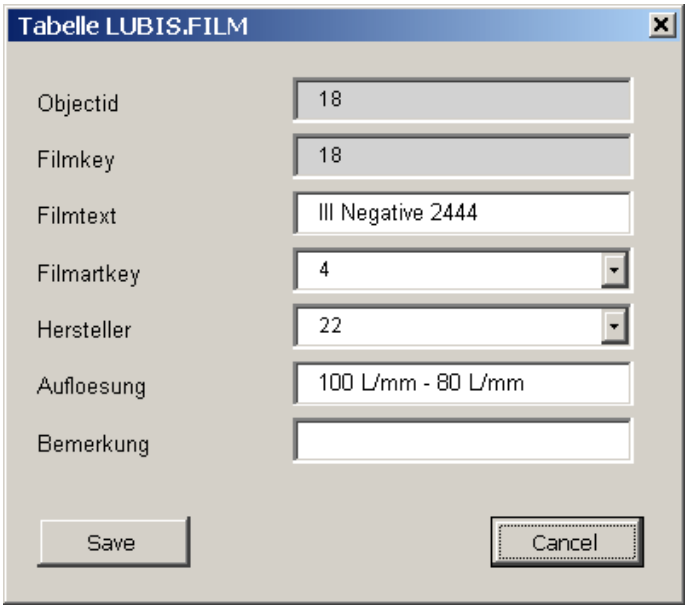


Figure 3: Completed form of the attribute table film

Input check

In VBA GUI elements are usually linked with event handling methods (e.g. after update, click, change, etc.), which respond to user inputs. For instance, input values may be checked for the correct data type, or logical dependences between several input fields may be examined. However, a restriction in VBA is that links between GUI elements and event handling methods must be established by corresponding GUI element name and event handling method name. For instance, the GUI element `cmdSave` has `cmdSave_Click()` as its clicking event handling method. Unfortunately, no dynamic coupling between GUI element and event handling is possible.
Due to this restriction and the fact that the tool described here creates the GUI elements dynamically at runtime, it is not possible to create corresponding code for event handling. We could not find a way for dynamic code generation as an alternative. Thus we have to pass on an immediate input examination. After clicking the save-button the appropriate event handling method is called. This method checks for each GUI element if the entered value conforms to the definition of the associated attribute in the database as follows:

- isNullable: An empty input is allowed.
- isNumeric: Input must be a numerical value.
- isDate: The attribute requires a date as input.

If an input value does not correspond to the attribute definition a corresponding error message is generated. If all inputs are correct the new dataset can be recorded.

Create dataset

For each attribute of the table the corresponding value of the GUI element is converted into the defined data type. The following data types are currently provided:

- esriFieldTypeSingle
- esriFieldTypeDouble
- esriFieldTypeSmallInteger
- esriFieldTypeInteger
- esriFieldTypeString
- esriFieldTypeDate

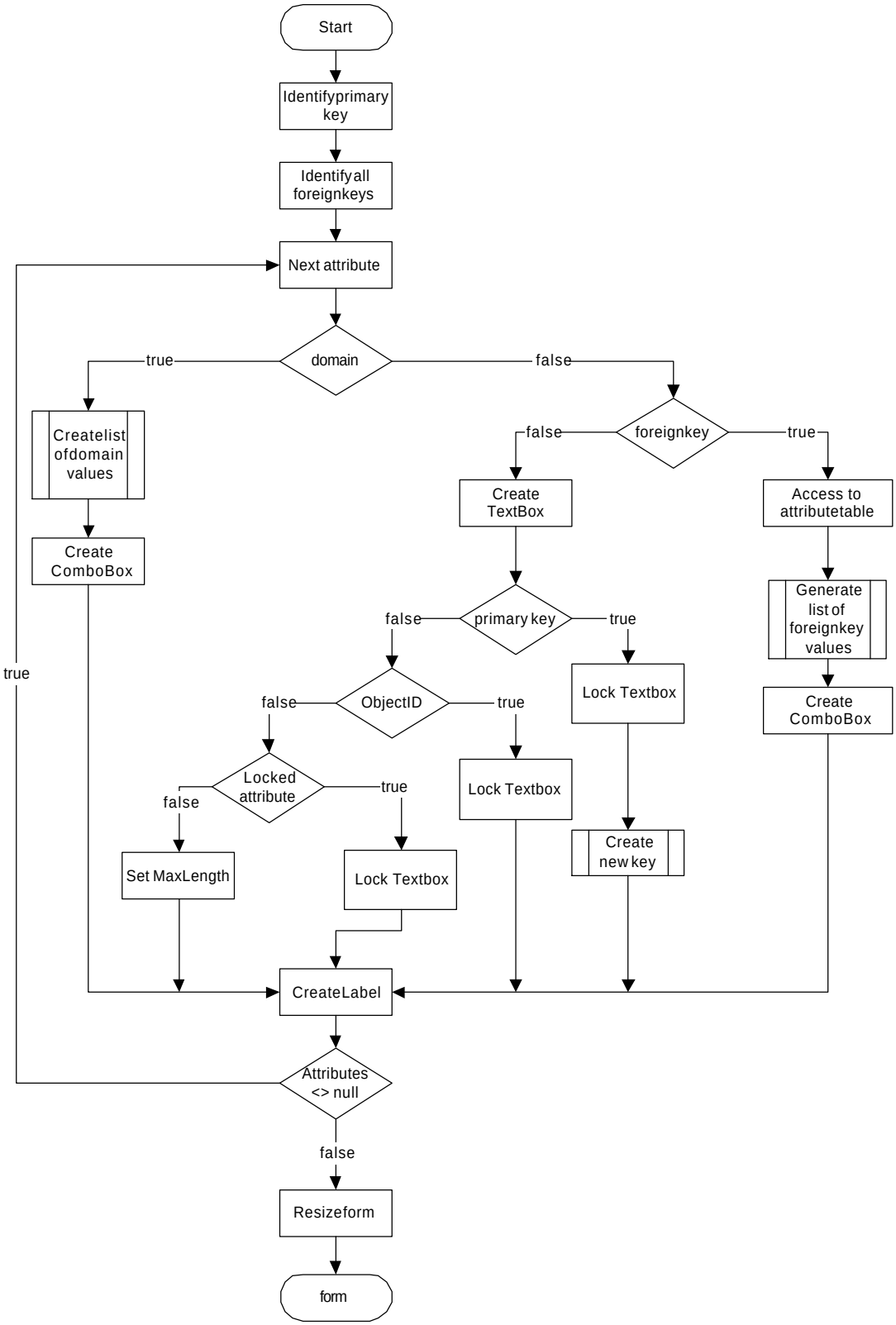
These are the only data types implemented since the database model of the system does not use any other types.

Conclusion

The presented tool has two particular features which are not supported by currently available ESRI standard tools:
First, the experienced user is able to create new records, to change and to delete recorded datasets without breaking database integrity rules by accident.
Second, we developed a new generic approach for dynamic creation of GUI elements which is only based on the definitions, constraints and values of the corresponding table. As a major advantage, changes of the table definitions, such as adding new attributes, has no influence on the base algorithm.
Experiences with this tool a very satisfying. The high user acceptance of the system confirms the user-friendliness of the system. In addition, the tool has worked faultless since being in operation.

Appendix

The complete algorithm for GUI generation as a flow chart:



References

Gehrig J. and W. Landolt 2002. An Aerial Photo Information System: Developing a Multi-User Application. *ESRI User Conference 2002 Proceedings. ESRI, 380 New York Street, Redlands, CA*. http://gis.esri.com/library/userconf/proc02/pap1057/p1057.htm

Höck M. and J. Manegold 2001. *ArcMap Programmierung mit VBA*. Eigenverlag. http://www.arcobjectsbuch.de

Lomax P. 1998. *VB & VBA in a Nutshell: The Language*. O'Reilly & Associates, Inc.

Author Information

Joerg Sparenborg
Swiss Federal Research Institute WSL
Section Landscape Inventories
Zürcherstrasse 111
CH-8903 Birmensdorf, Switzerland
phone: +41 1 739 23 77
fax: +41 1 739 22 15
email: joerg.sparenborg@wsl.ch
web: http://www.wsl.ch