Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

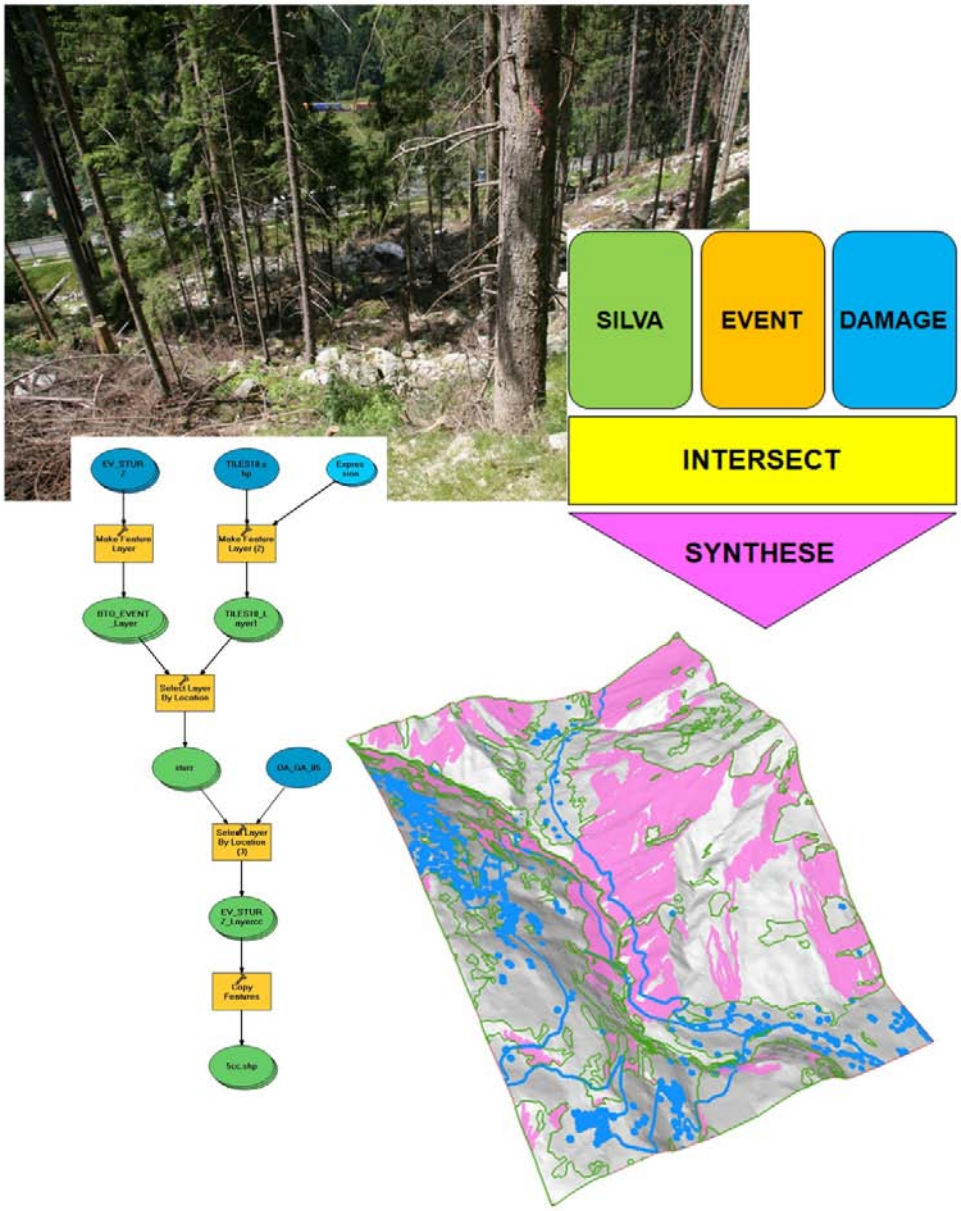
Eidgenössisches Departement für
Umwelt, Verkehr, Energie und Kommunikation UVEK

Bundesamt für Umwelt BAFU
Abteilung Gefahrenprävention

Januar 2013 / LOS

SilvaProtect-CH: **GIS Handbuch**

Anhang 3



(Photo S. Losey, 2011)

Inhalt

1 Einführung

Mit diesem Handbuch sollen die folgenden Ziele erfüllt werden: (1) Klarheit bei der GIS-Berechnung für das Projekt SilvaProtect-CH, (2) Hilfe und Grundlage für künftige Berechnungen von Schutzwald- und Schadenpotenzialindex (SWI und SPI) und (3) Methodik für weitere grosse GIS-Projekte.

Die Struktur von SilvaProtect-CH ist im Kapitel 2.2 des Schlussberichtes beschrieben. Die verschiedenen Kapitel des GIS-Handbuchs haben eine direkte Beziehung zu den Hauptmodulen des Projekts SilvaProtect-CH (Abbildung 1).

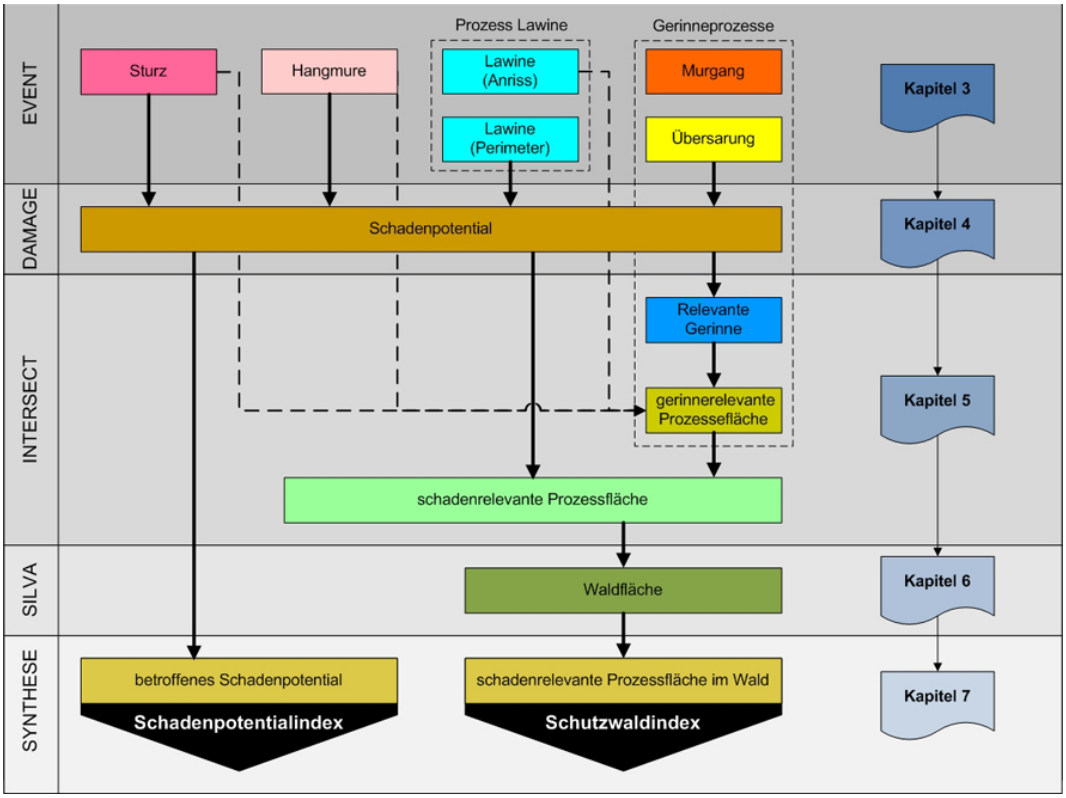


Abbildung 1: Schematische Darstellung der einzelnen Module von SilvaProtect-CH mit Bezug zu den Kapiteln des GIS-Handbuches.

Zusätzlich zu den in Abbildung 1 dargestellten Kapiteln gibt Kapitel 2 Auskunft über allgemeine Grundlagen wie die benutzte Software und die Struktur der georeferenzierten Daten.

2 Information zur EDV und Datenstruktur

2.1 Hardware und Software

Alle Modellierung wurden auf einer lokalen Workstation erstellt, da der Datenumfang eine Bearbeitung via Server nicht zuliess. Die Daten auf der Workstation wurden laufend auf einer zusätzlichen externen Festplatte gesichert.

Im Rahmen des Projekts SilvaProtect-CH wurde die folgende Software gebraucht:

- ArcGIS 9.3: für die Modellierung des Projekts
- ModelBuilder: Geoverarbeitung für die Modellierung
- Python 2.5: Programmiersprache für die Lawinenmodellierung

Es wurde folgende Ordnerstruktur für die Datenablage verwendet

- D:\Grunddaten: Topographische Karte 25'000
- D:\SilvaProtect: Projekt SilvaProtect
    - ADMIN: Administrative Informationen zum Projekt
    - BASIC: Inputdaten (Kapitel 2.6chapitre 0)
    - DAMAGE: Daten und Modellierung des Schadenpotenzials (Kapitel 4)
    - EVENT: Gefahrenprozesse (Kapitel 3)
    - INTERSECT: Relevante Prozesse (Kapitel 5)
    - SILVA: Waldfläche (Kapitel 6)
    - SYNTHESE: Resultate (Kapitel 7)
    - TOOLBOX: Modelle für die Geoverarbeitung mit ModelBuilder (Anhang 9.1)
- D:\Temp Silva: Temporäre Daten für die Berechnung der Zwischenresultate
- D:\Workspace: Temporäre Daten, die nach der Geoverarbeitung direkt gelöscht werden.

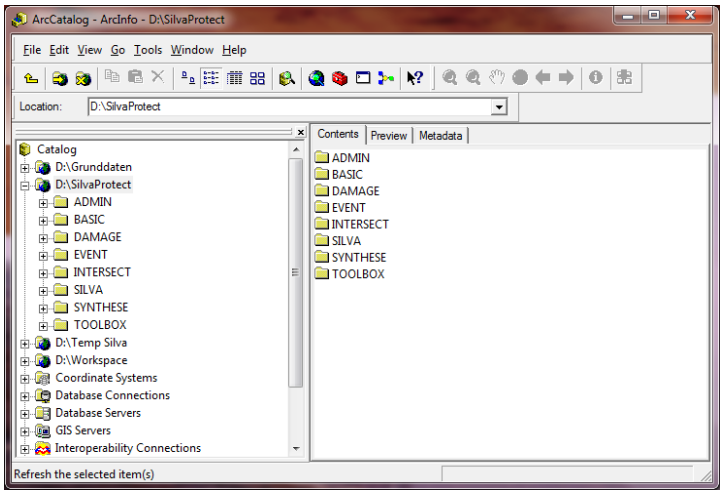


Abbildung 2: Speicherort der Daten.

2.2 Management von grossen Datenmengen

Durch die Datenmenge von SilvaProtect-CH (Sturz: 9.3 Mio Trajektorien, Dateigrösse 1.2 GB; Hangmure: 47.6 Mio Trajektorien, Dateigrösse 6.5 GB) ist die Grenze des virtuellen Speichers schnell erreicht. Deswegen wurde die ganze Schweiz in Kacheln eingeteilt, welche bei der Modellierung nacheinander abgearbeitet werden. Diese Einteilung wurde unter Berücksichtigung der folgenden Kriterien gemacht:

- Verschiedene Kachelgrösse — Erlaubt in Abhängigkeit der Datenmenge ein flexibles Vorgehen (Wahl einer anderen Grösse bei Bedarf)
- Schachtelung der Kacheln — Erlaubt die Teilung von Kacheln, falls die Datenmenge innerhalb einer Kachel zu gross ist
- Saubere Nummerierung — Erlaubt eine saubere und nachvollziehbare Dokumentation der Beziehung zwischen grösseren und kleineren Kacheln.

Die Schweiz wurde somit primär in 49 Hauptkacheln mit einer Kantenlänge von 40 Kilometer aufgeteilt und die Kacheln von Nord-Osten nach Süd-Westen durchnummeriert. (Abbildung 3).

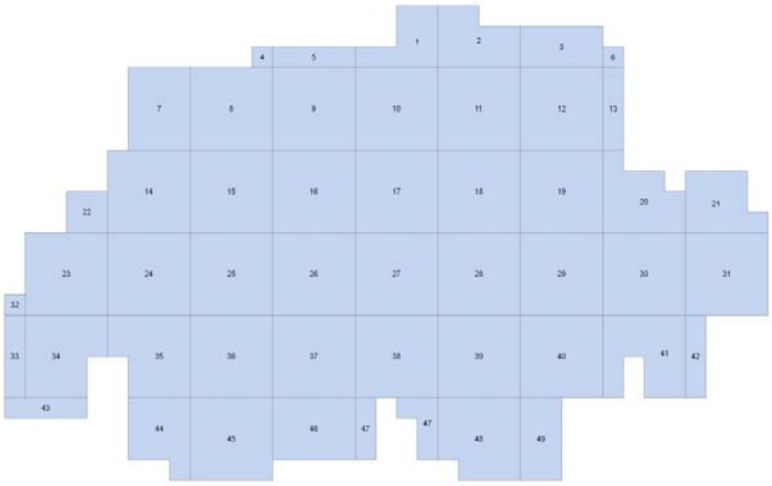


Abbildung 3: Die 49 Hauptkachel der Schweiz.

Jede Hauptkachel wurde dann in 4 Kacheln à 20 Kilometer Kantenlänge und in 16 Kacheln à 10 Kilometer Kantenlänge aufgeteilt (Abb. 4).

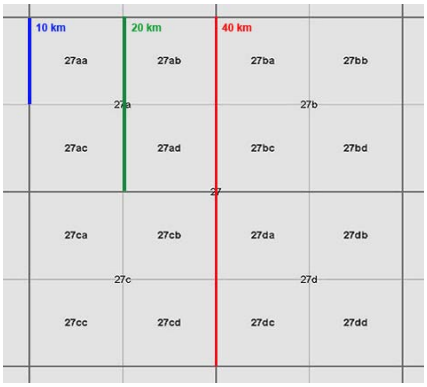


Abbildung 4: Aufteilung der Hauptkacheln in weitere Kacheln.

Diese Methode erleichtert auch die Programmierung mit dem ModelBuilder.

2.3 Umwandlung der Trajektorien zu Flächen

Selbst wenn die kleinsten Kachel eine Kantenlänge von 10 Kilometern haben, ist es oft möglich, dass mehr als 200'000 Trajektorien in dieser Kachel vorhanden sind. Das GIS-Tool "Dissolve" kann so viele Daten nicht behandeln. Um diese Daten von Trajektorien in Polygone umzuwandeln, ist es daher einfacher und schneller mit Rastern zu arbeiten.

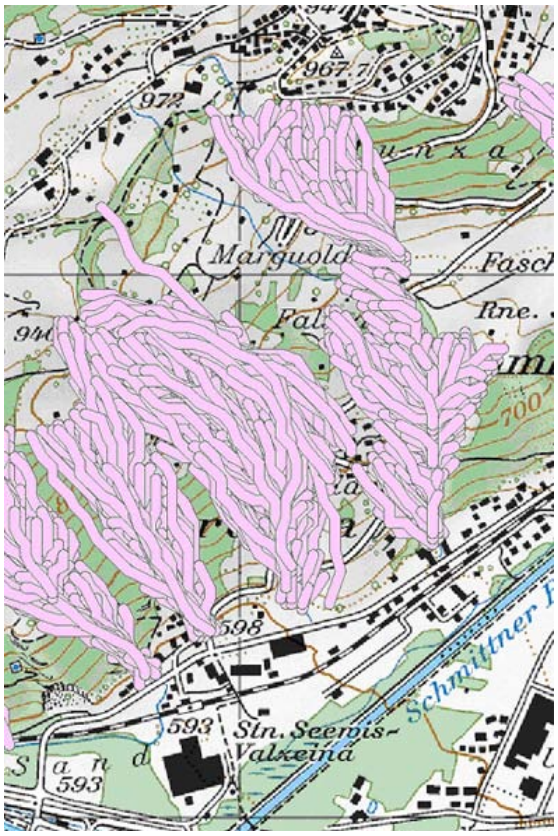


Abbildung 5: Trajektorien von Hangmuren mit einem Puffer von 10m

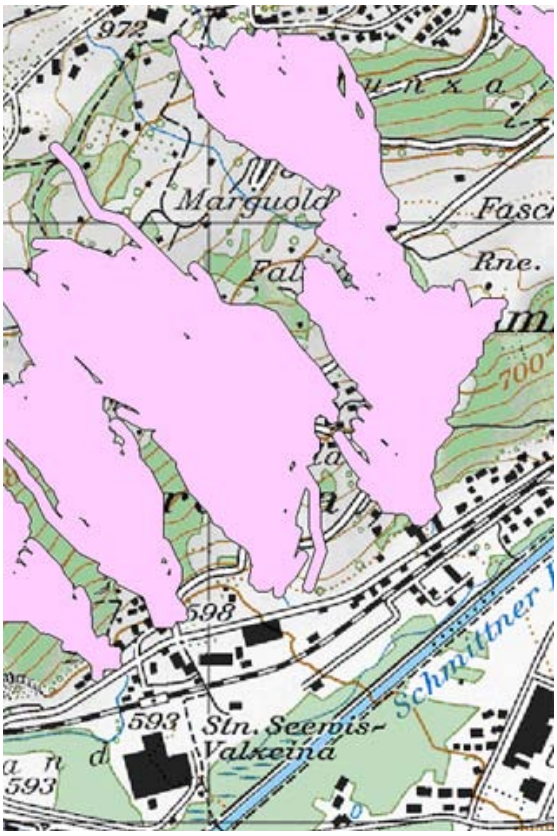
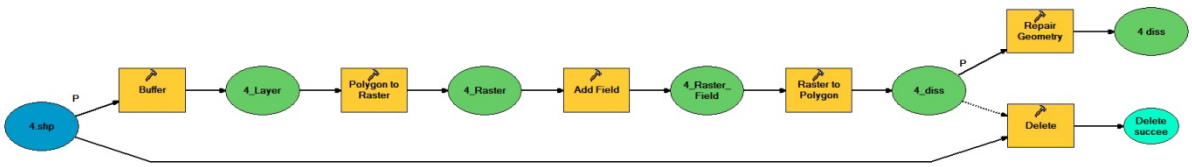


Abbildung 6: Polygon aus zusammengefassten Trajektorien.

Im Rahmen des Projekts SilvaProtect-CH wurden die Trajektorien immer in Polygone umgewandelt; die Methodik ist unten beschrieben.



Modell 1: Umwandlung von Linien in Polygone.

Beschreibung

- **Buffer** Distance: 10 m
- **Polygon to Raster** Value Field: FID; Output Raster Dataset: "*Format .img wegnehmen; damit wird eine Tabelle mit Attributen kreiert,* Cellsize: 1
- **Add Field** Field Name: Diss; Field Type: short
- **Raster to Polygon** Field: Diss; Simplify Polygon checked

Die beiden „P“ im Modell 1 erlauben es, Input- und Output-Daten zu definieren. Damit können alle berechneten Kacheln, unabhängig von ihrer Lage in der Schweiz, in dieses Modell eingefügt werden.

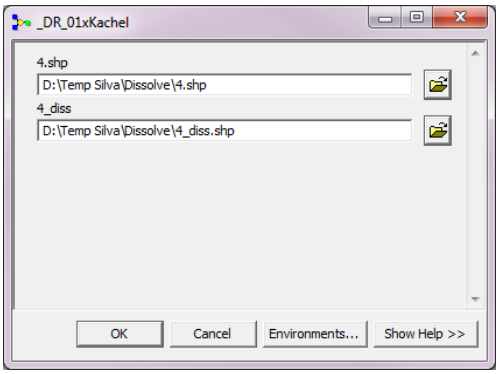


Abbildung 7: Parameter im Modell 1.

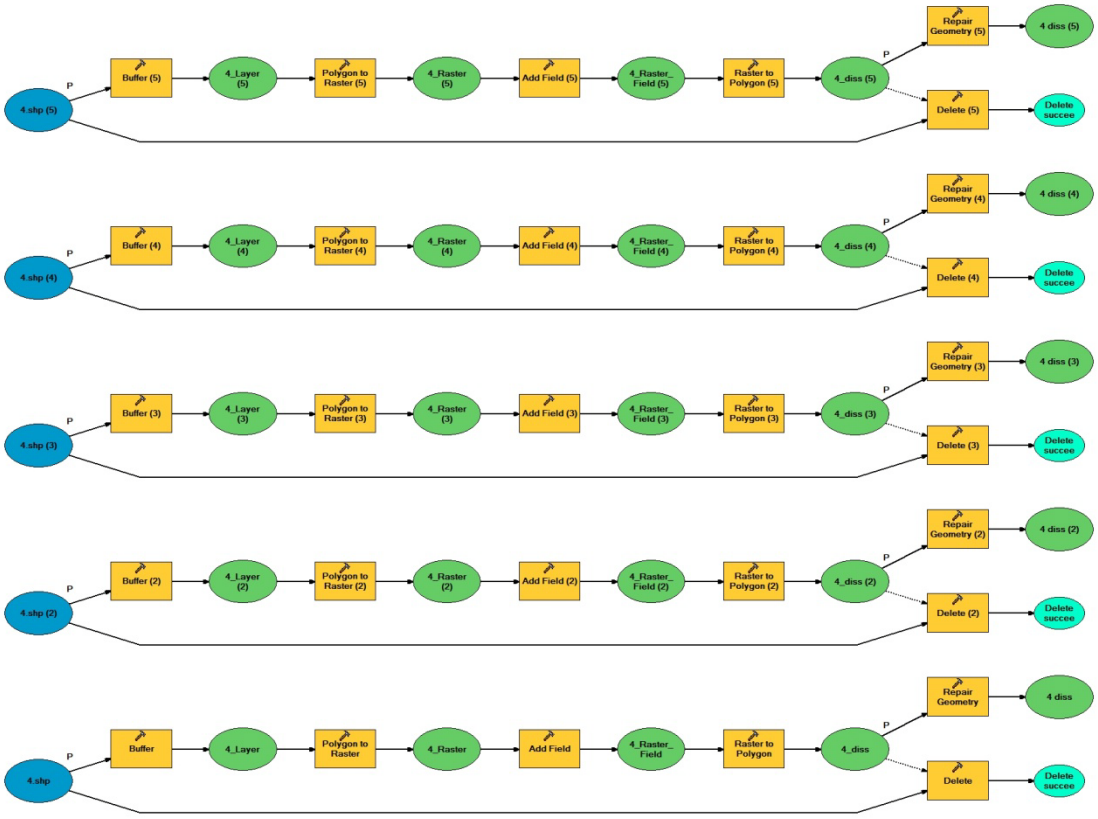
2.4 Ablauf für lange Modellierungen

Wenn die Modellierung die ganze Schweiz abdecken muss, ist es nötig, dass alle einzelnen Prozesse nacheinander in einem einzigen Prozess implementiert werden.

Drei Methoden werden im Projekt SilvaProtect-CH verwendet, um diese langen Modellierungen durchzuführen.

Methode 1: Mehrere Prozesse in einem Modell

Diese Methode (Modell 2) eignet sich dann, wenn die Berechnungen unabhängig von der Lage in der Schweiz sind. Mit dem Modellparameter (Abbildung 7) können mehrere „Input“ (P) und „Output“ (P) im Modell 1 eingefügt werden. Dazu können die zwischengespeicherten Daten denselben Namen haben; dies vereinfacht die Modellierung.



Modell 2: Mehrere Prozesse in einem Model.

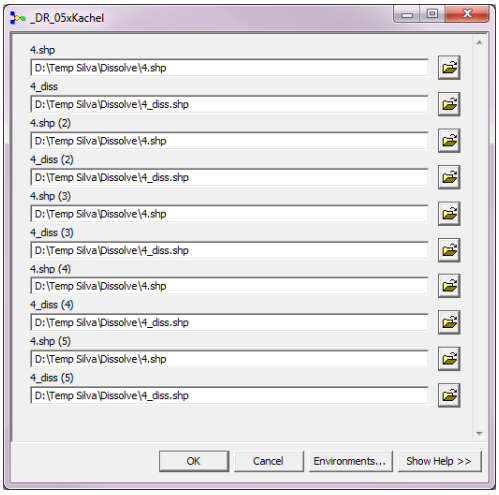


Abbildung 8: Mehrere Modellparameter im Modell 2.

Methode 2: Modell mit „a list of value“

In den Einstellungen einer Variablen kann „a list of values“ anstatt „a single value“ gewählt werden (Abbildung 9). Mit diesem Modell 3 kann man nicht nur eine sondern gleich mehrere Kacheln zusammen berechnen (Abbildung 10). Die Zwischenresultate müssen dazu aber unbedingt einen anderen Namen haben (Abbildung 11).

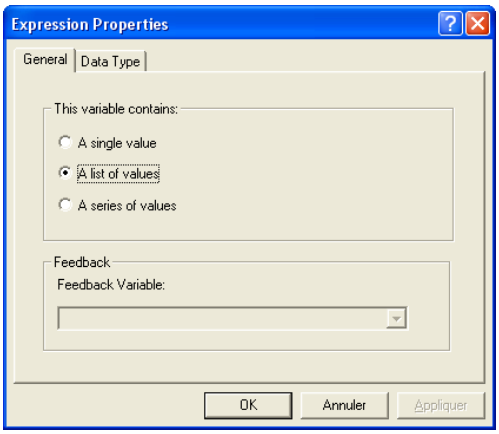
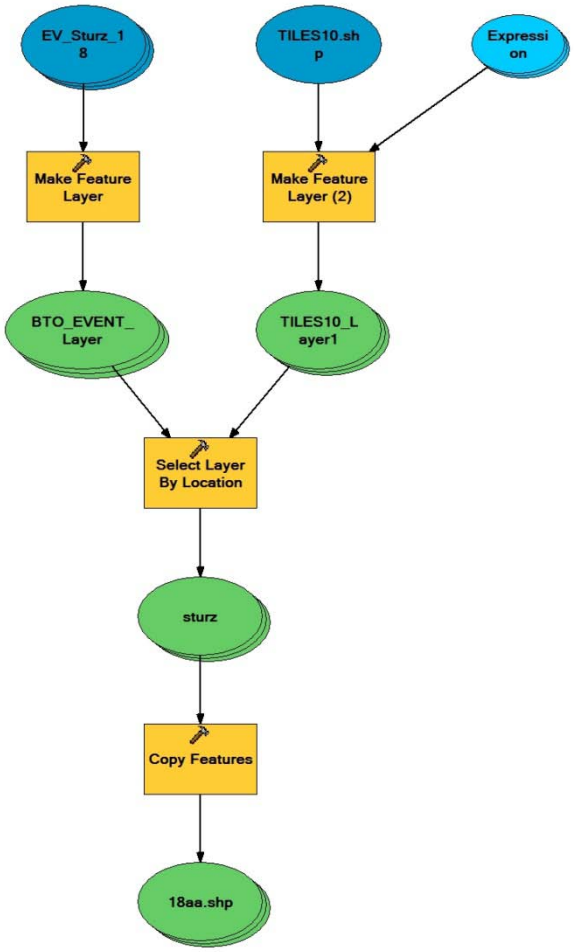


Abbildung 9: Eigenschaften eines Modells, um ein "liste of value" zu berechnen.

Wenn ein solches Modell aufgebaut ist, müssen alle Shapefiles mit „list of value“ angepasst werden.



Modell 3: Beispiel eines Prozesses mit "list of value".

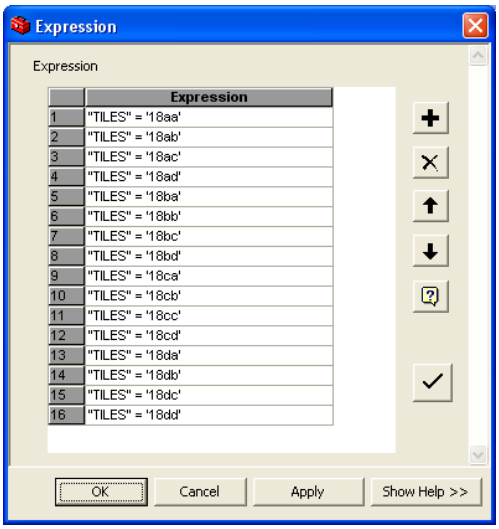


Abbildung 10: Eigenschaften von "Expression" für das Modell 3.

Mit diesem Modelltyp und der Einteilung der Schweiz in Kacheln kann ein Modell für eine Kachel schnell auf andere Kacheln angepasst werden. So kann beispielsweise im Modell der Abbildung 10 die Kachel 18 ohne Problem durch die Kachel 19 ersetzt werden.

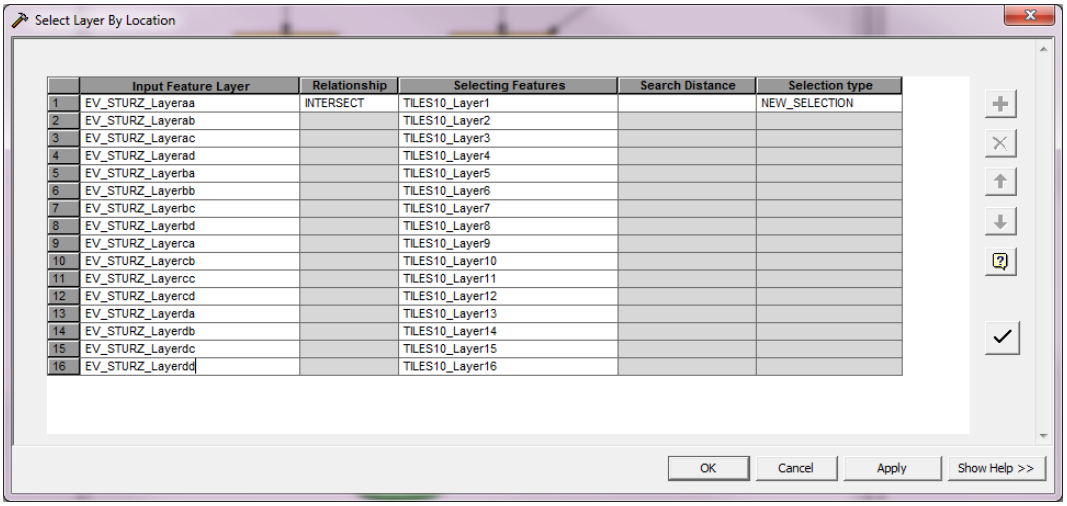


Abbildung 11: Eigenschaften von " Select by location" für das Modell 3.

Die Zwischenresultate, die nicht gespeichert sind, müssen innerhalb des Modells mit anderen Namen gespeichert werden. Wenn das Modell hingegen mehrmals benutzt wird, braucht man die Zwischenresultate nicht umzubenennen, weil sie nicht gespeichert werden.

Methode 3: Mehrere Prozesse in einen Hauptprozess implementieren

Diese Lösung (Modell 4) erlaubt, mehrere vorgängig erstellte Modelle zu berechnen, sofern die Lage der eingefügten Daten klar definiert ist. . Damit kann jede Kachel der Schweiz (Abbildung 3) und damit die ganze Schweiz in einem Zug nacheinander berechnet werden. Das folgende Modellbeispiel berechnet die Hangmurtrajektorien, welche auf ein Schadenpotenzial treffen. Diese Berechnung kann länger als 50 Stunden dauern.

Modell 4: Verschiedene Modelle sind in einem Hauptmodell implementiert.

2.5 Farbcode im GIS-Handbuch

Verschiedene Farben werden im GIS-Handbuch wie folgt verwendet:

- Blau Name für die Vektordaten, die im Modell als Input gelten
- Grün Name für die Vektordaten, die im Modell als Output gelten
- Rot Name des Modells, das mit ModelBuilder erarbeitet ist.

2.6 Grunddaten (BASIC)

Diese Grunddaten wurden für die verschiedenen Modellierungen gebraucht. Es gibt drei Arten von Daten:

- Die politischen Daten: Grenzen der Schweiz und der Kantone
- Die Daten für die Grösse der Modellierung: Kacheln von 10, 20 und 40 km Kantenlänge und
- Die thematischen Daten: Lawine und Übersarung

Die Schweiz (**CH.shp**):

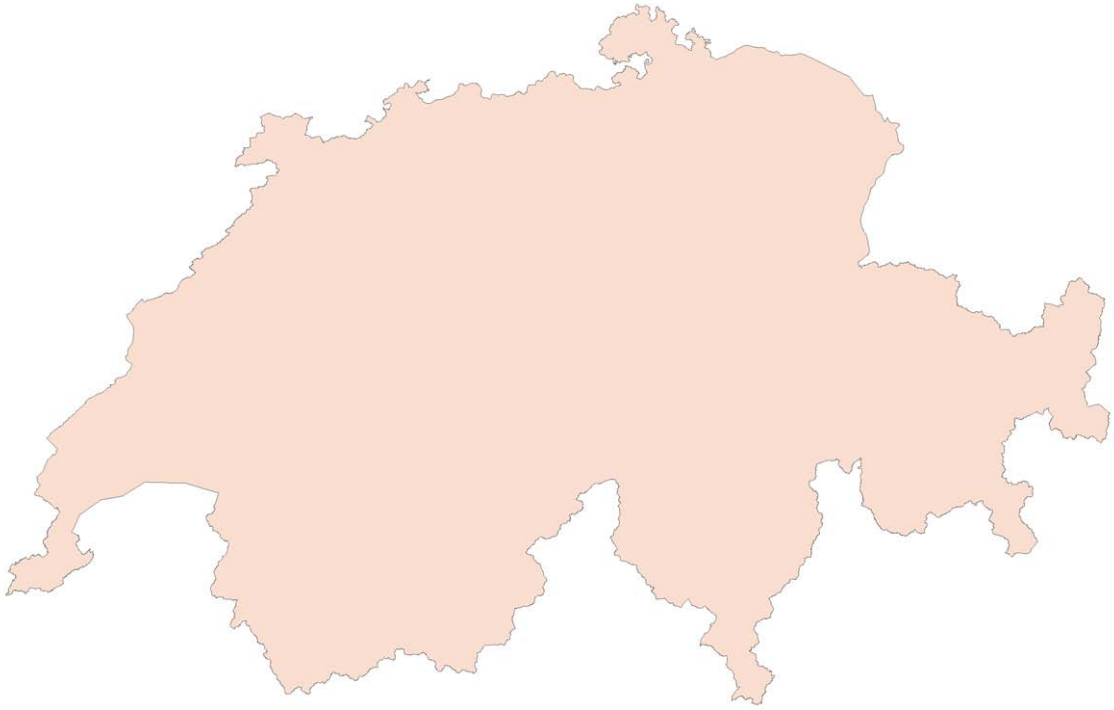


Abbildung 12: Shape für die ganze Schweiz.

Die Kantone: (**Kantone.shp**):

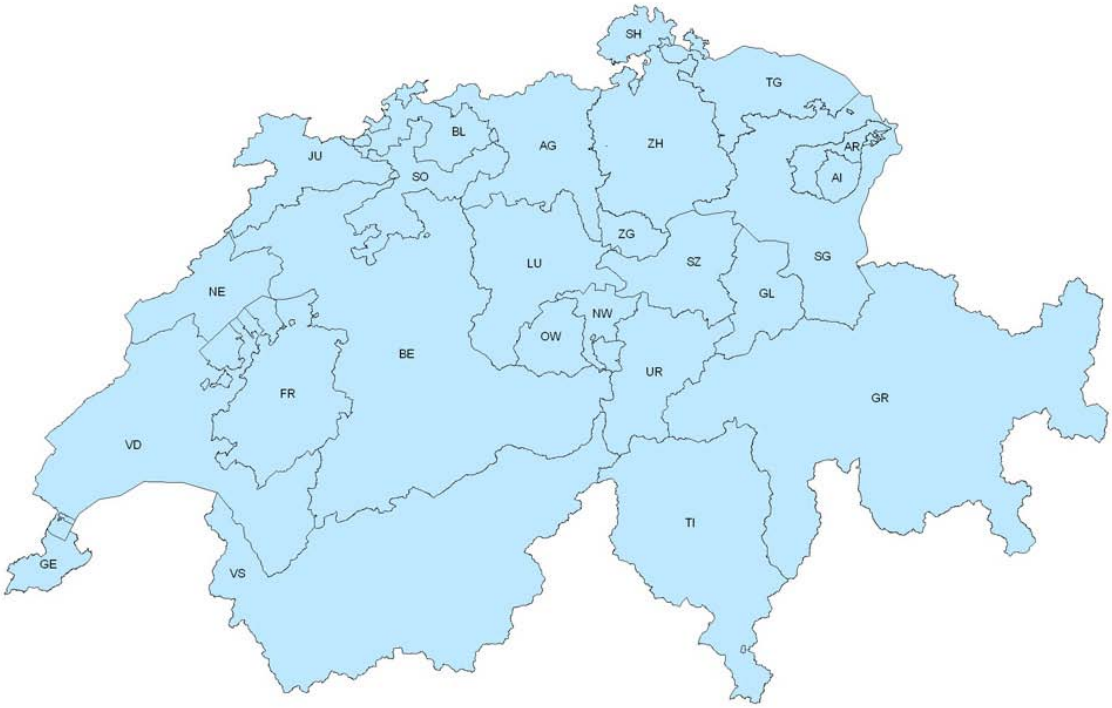


Abbildung 13: Shape für die Kantone.

Einteilung der Schweiz in 10 km Kacheln (**TILES10.shp**):

Abbildung 14: Shape für die 10 km Kacheln.

Einteilung der Schweiz in 20 km Kacheln (**TILES20.shp**):

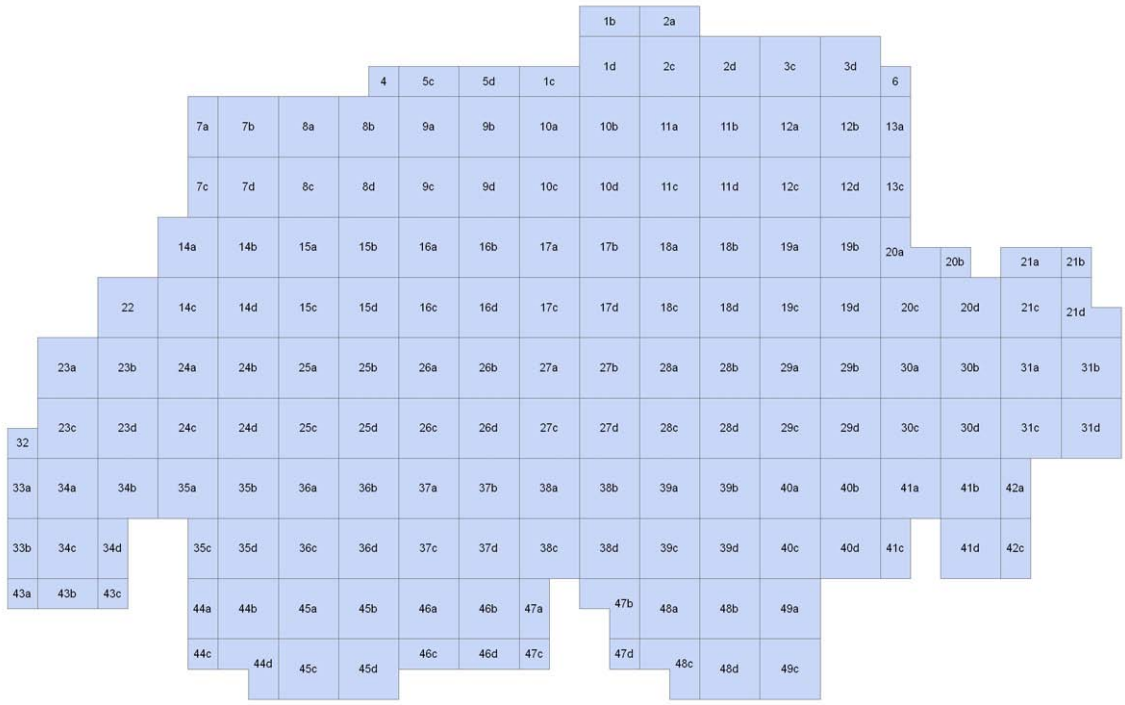


Abbildung 15: Shape für die 20 km Kacheln.

Einteilung der Schweiz in 40 km Kacheln (**TILES40.shp**):

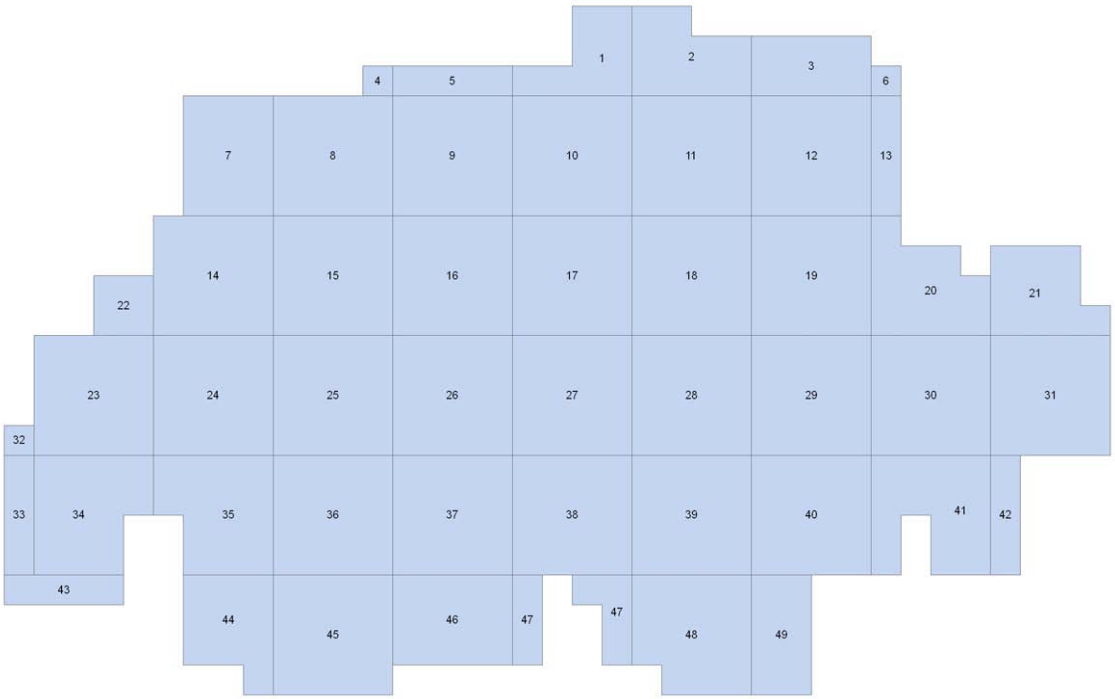


Abbildung 16: Shape für die 40 km Kacheln.

Die Lawinengebiete für die Lawinenmodellierung (**BA_Lawine_Gebiete.shp**):

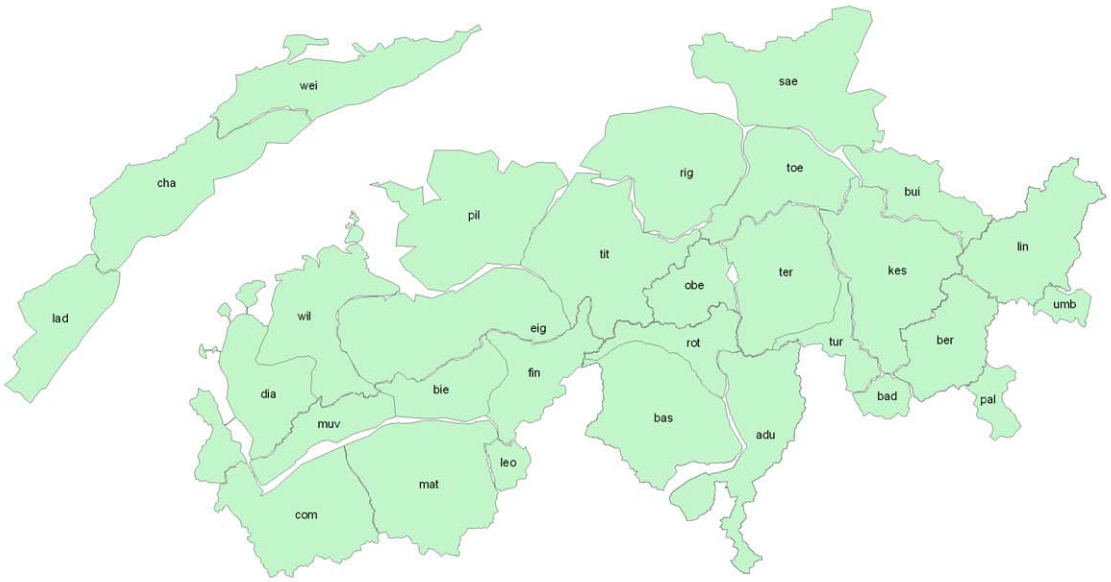


Abbildung 17: Berechnungsgebiete für die Lawine.

Die Übersarungsgebiete für die Modellierung der Übersarungen (**Übersarungsgebiete.shp**):

Abbildung 18: Berechnungsgebiete für die Übersarungsflächen.

3 Gefahrenprozesse (EVENT)

Folgende Gefahrenprozesse wurden berücksichtigt und modelliert:

- Lawine
- Sturzprozesse
- Rutsch/Hangmure
- Gerinneprozesse

3.1 Lawine

Jede modellierte Lawine besteht aus zwei Komponenten, dem Perimeter (Umriss der Lawine) und dem Entstehungsgebiet (Start der Lawine, wo der Wald eine Schutzwirkung leisten kann).

Für die Berechnung des Gefahrenprozesses Lawine wurde die Schweiz in 30 Lawinengebiete geteilt (Abbildung 17). Pro Gebiet (xxx) braucht man für die Berechnung die folgenden Daten:

- xxxn_s_ph_r: Perimeter für kleine Lawine
- xxxn_s_rel_r: Entstehungsgebiet für kleine Lawine
- xxxn_m_ph_r: Perimeter für mittlere Lawine
- xxxn_m_rel_r: Entstehungsgebiet für mittlere Lawine
- xxxn_l_ph_r: Perimeter für grosse Lawine
- xxxn_l_ph_r: Entstehungsgebiet für grosse Lawine
- xxxn_l_cap / point: Punkte für die Berechnung von grossen Lawinen

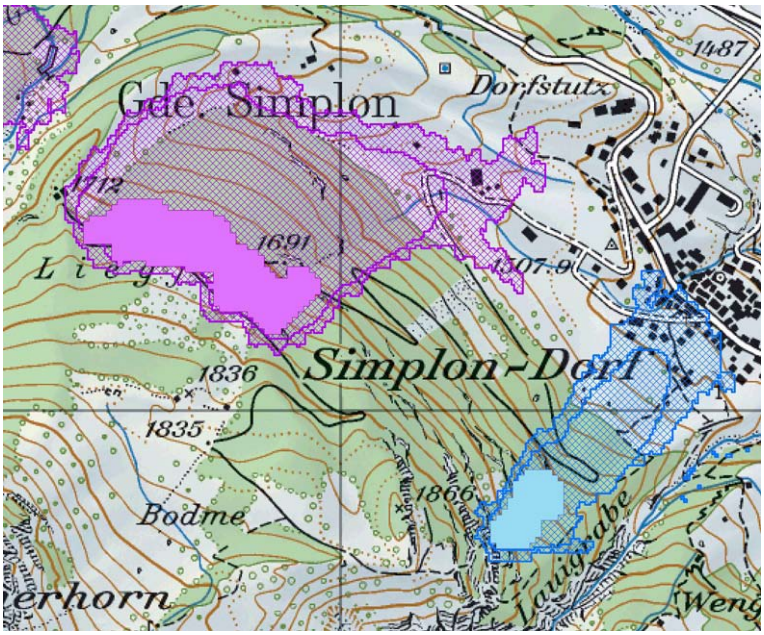


Abbildung 19: Perimeter einer kleine Lawine (blaue Schraffur) mit ihrem Entstehungsgebiet (blau) und einer mittleren Lawine (violette Schraffur) mit ihrem Entstehungsgebiet (violett).

Kürzel	Gebietsname	Kürzel	Gebietsname
adu	Piz Adula	mat	Matterhorn
bad	Piz Badile	muv	Muverain
bas	Basodino	obe	Oberalpstock
ber	Piz Bernina	pal	Piz Palü
bie	Bietschhorn	pil	Pilatus
bui	Piz Buin	rig	Rigi
cha	Chasseral	rot	Pizzo Rotondo
com	Grand Combin	sae	Säntis
dia	Les Diablerets	ter	Piz Terri
eig	Eiger	tit	Titlis
fin	Finsteraarhorn	toe	Tödi
kes	Piz Kesch	tur	Piz Turba
lad	La Dôle	umb	Pizzo Umbreil
leo	Monte Leone	wie	Weissenstein (JU)
lin	Piz Linard	wil	Wilerhorn

Tabelle 1: Abkürzungen und Name den einzelnen Berechnungsgebiete für die Lawine.

3.2 Sturz

Wald kann die kinetische Energie von fallenden Steinen und Blöcken stark reduzieren. Die Modellierung der Sturztrajektorien über das gesamte Gebiet der Schweiz ist im Anhang 1 beschrieben. Es wurden 9.3 Mio. Sturz-Trajektorien (EV_Sturz) berechnet und abgespeichert. Diese Trajektorien wurden in 49 Kacheln à 40 km Kantenlänge eingeteilt (EV_Sturz_01 bis EV_Sturz_49, siehe Abbildung 3), um die Berechnungsgeschwindigkeit zu erhöhen.

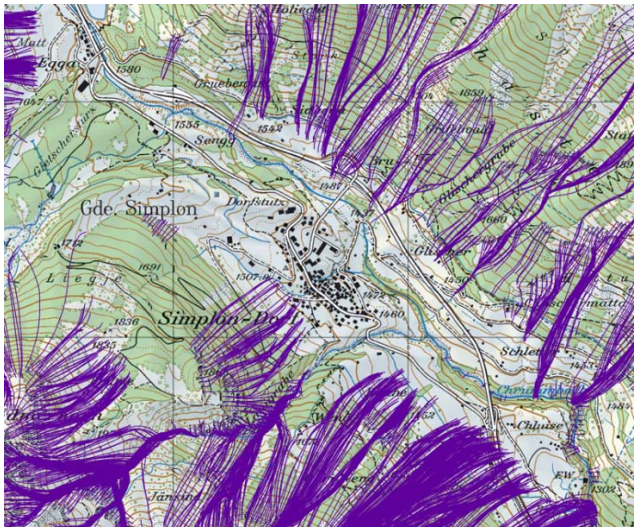


Abbildung 20: Sturztrajektorien (violett)

3.3 Rutsch / Hangmure

Dank den Baumwurzeln kann der Wald Rutschungen und Hangmuren verhindern. Die Modellierung der Hangmurentrajektorien über das gesamte Gebiet der Schweiz ist im Anhang 1 beschrieben. Es wurden 47.6 Mio. Sturz-Trajektorien (EV_Hangmure) berechnet und abgespeichert. Diese Trajektorien wurden in 49 Kacheln à 40 km Kantenlänge eingeteilt (EV_Hangmure_01 bis EV_Hangmure_49, siehe Abbildung 3), um die Berechnungsgeschwindigkeit zu erhöhen.

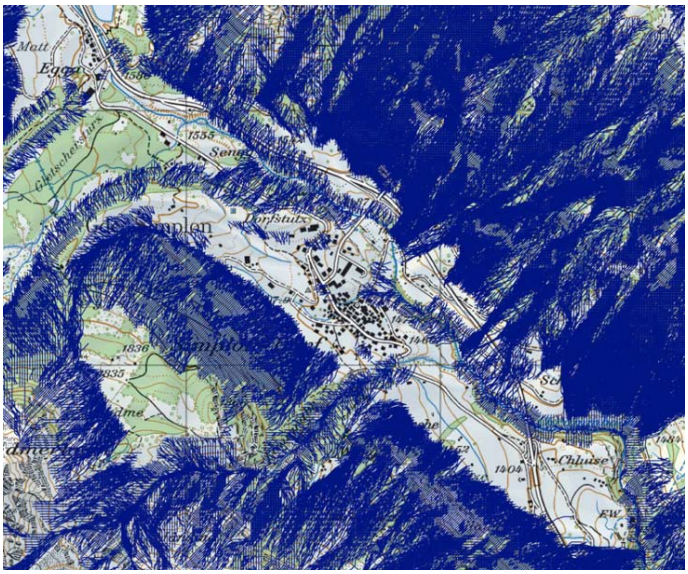


Abbildung 21: Hangmurentrajektorien (blau).

3.4 Gerinneprozesse

Die Wälder verhindern, dass die oben beschriebenen Prozesse Material in die relevanten Gerinne bringen. Um die relevanten Gerinne zu bestimmen, werden die georeferenzierten Daten Übersarung und Murgang benötigt.

Übersarungen

Wenn Material aus einem Gerinne tritt, gibt es dafür immer einen Startpunkt sowie eine Übersarungsfläche. Diese beiden Komponenten wurden pro Übersarungsgebiet (Abbildung 18) modelliert:

- EV_Uebersarung_xxx: Übersarungsfläche
- EV_Uebersarung_start_xxx: Startpunkte für die Übersarungen

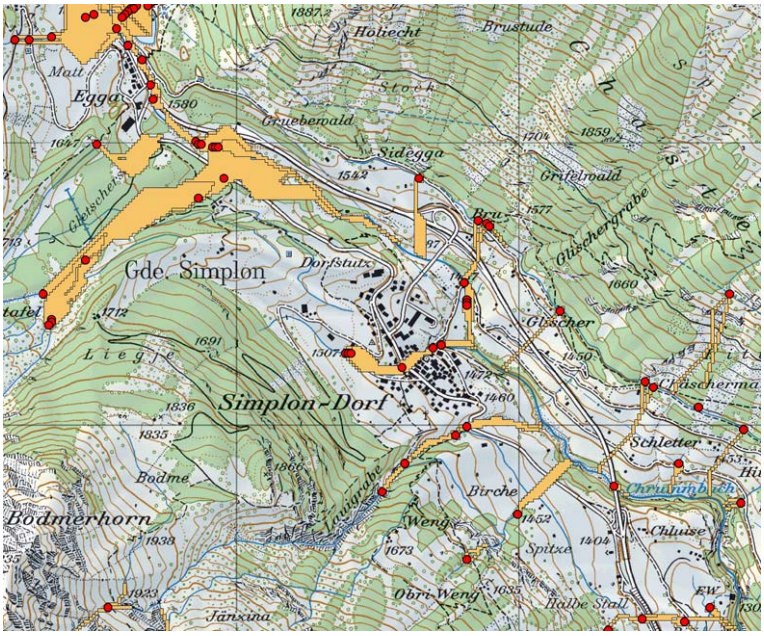


Abbildung 22: Übersarungsflächen (beige) und Startpunkte (rot).

Kürzel	Gebietsname
aa	Aare Alpin
in	Inn
ju	Jura
mn	Mittelland-Nord
ms	Mittelland-Süd
ra	Rhone-Alpin
rna	Rhein-Alpin
rla	Reuss/Linth Alpin
rm	Rhone Mittelland
rnm	Rhein-Mitte
ti	Ticino

Tabelle 2: Abkürzungen und Name der einzelnen Übersarungsgebiete für die Modellierung.

Murgang

Die Modellierung der Murgangtrajektorien über das gesamte Gebiet der Schweiz ist im Anhang 1 beschrieben. Es wurden 6.7 Mio. Murgangtrajektorien (EV_Murgang) berechnet und abgespeichert. Diese Trajektorien sind in 49 Kacheln à 40 km Kantenlänge eingeteilt (EV_Murgang_01 bis EV_Murgang_49, siehe Abbildung 3), um die Berechnungsgeschwindigkeit zu erhöhen.

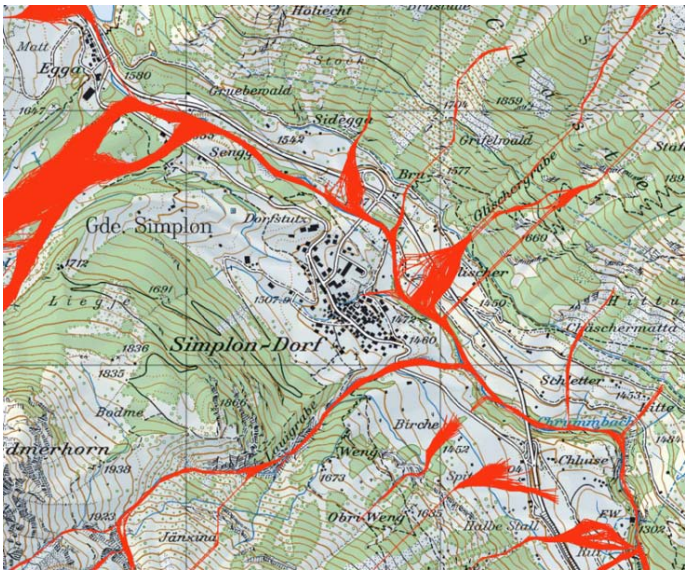


Abbildung 23: Murgangtrajektorien (rot).

4 Schadenpotenzial (DAMAGE)

Wie aus der Definition des Schutzwaldes hervorgeht, ist ein Wald ein Schutzwald, wenn er ein anerkanntes Schadenpotenzial gegen eine bestehende Naturgefahr schützen oder die damit verbundenen Risiken reduzieren kann. Im Kapitel 3 wurden die anerkannten Gefahrenprozesse definiert und georeferenziert. In Kapitel 4 wird nun das Schadenpotenzial beschreiben. Das Schadenpotenzial ist in verschiedene Kategorien eingeteilt:

- Eisenbahnnetz
- Anlagen
- Gebäude
- Strassennetz

Für die Berechnung des Schutzwaldindex wurden alle Schadenpotenzialkategorien als gleichwertig betrachtet und nicht gewichtet.

Im Gegenteil dazu wurden die Schadenpotenzialkategorien **für die Berechnung des Schadenpotenzialindex** unterschiedlich gewichtet:

Gruppierung	**Beschreibung**	**Gewichtung**
Gruppe 1	Dauernde oder beinahe dauernde Präsenz von Menschen	10
Gruppe 2	Zeitweiser Aufenthalt bei hoher Präsenzwahrscheinlichkeit	5
Gruppe 3	Zeitweiser Aufenthalt bei niedriger Präsenzwahrscheinlichkeit	3
Gruppe 4	Versorgungsstörungen und/oder hohe Sachschäden	1
Gruppe 5	Restliche Objekte	0

Tabelle 3: Gewichtung des Schadenpotenzials für die Berechnung des SPI.

Bei überlappenden Elementen wurde im überlappenden Bereich nur das Element mit der höheren Gewichtung bei der Berechnung des Schadenpotenzialindex berücksichtigt.

Mit dem Suffix **swi** am Ende eines Shapefiles wurden Dateien markiert, die für die Berechnung des Schutzwaldindexes benutzt wurden.

Die folgenden Vektordaten werdenzur Modellierung des Schadenpotenzials benötigt und werden nach Bedarf aktualisiert. Die Daten sind unter **D:\SilvaProtect\DAMAGE\INPUT** auf der Workstation gespeichert.

Dateiname	Jahr	Quelle
bn97hn3	1997	BFS
DA_Betriebszaehlung01	2008	BFS
DA_Volkszaehlung00	2000	BFS
GSM_Sendeanlagen	2008	BAFU, Sektion Nichtionisierende Strahlung
str_25_arc	2008	Swisstopo
V25_anl_polygon	2008	Swisstopo
V25_eis_arc	2008	Swisstopo
V25_eob_arc	2008	Swisstopo
V25_eob_point	2008	Swisstopo
V25_geb_polygon	2008	Swisstopo
V25_gwn_arc	2008	Swisstopo
V25_pri_polygon	2008	Swisstopo

Tabelle 4: Vektordaten zur Modellierung des Schadenpotenzials.

Die zur Geoverarbeitung benötigten Modelle sind in einer Toolbox SiPro 93 DAMAGE unter **D:\SilvaProtect\TOOLBOX** gespeichert.

4.1 Eisenbahnnetz

Modell:	Eisenbahn GA (Toolbox: SiPro 93 DAMAGE)
Input:	V25_eis_arc.
Output:	DA_Eisenbahn_GA_G1, DA_Eisenbahn_GA_G3, DA_Eisenbahn_GA_G5 und DA_Eisenbahn_swi.

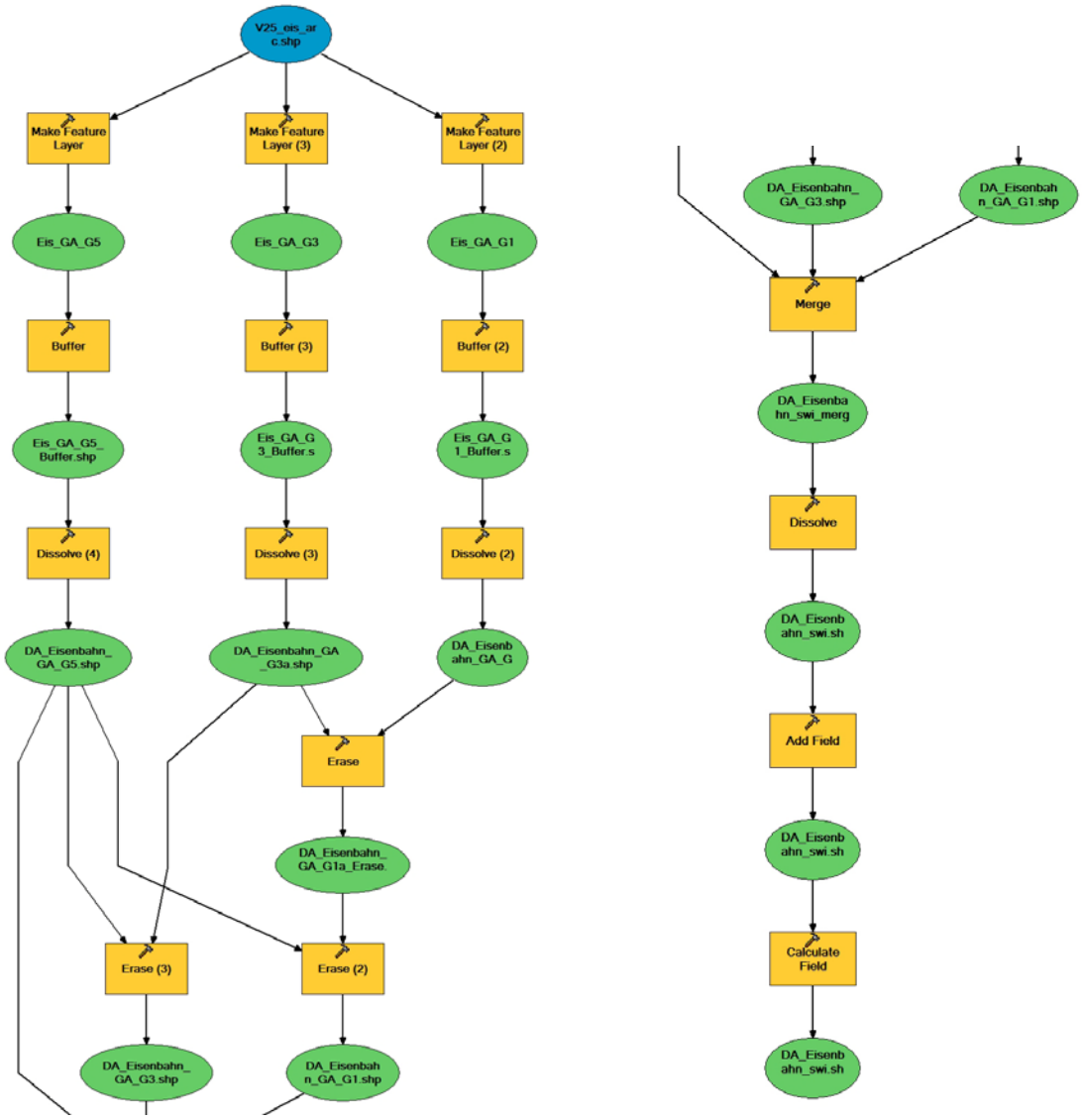
Beschreibung des Modells:

Das Modell unterscheidet das Eisenbahnnetz nach drei Gewichtungen:

- **Gewichtung 1**: Versorgungsstörungen und/oder hohe Sachschäden
  - Objekte: Industriegeleise
  - Bemerkung: kein Tunnel

- **Gewichtung 3**: Zeitweiser Aufenthalt bei niedriger Präsenzwahrscheinlichkeit
  - Objekte: Güterbahn, Normalspurbahn und Schmalspurbahn eingleisig
  - Bemerkung: kein Tunnel

- **Gewichtung 5**: Zeitweiser Aufenthalt bei hoher Präsenzwahrscheinlichkeit
  - Objekte: Normalspurbahn und Schmalspurbahn mehrgleisig
  - Bemerkung: kein Tunnel



Modell 5: Eisenbahnnetz.

Beschreibung:

GA G1

- **V25_eis_arc**: **Make Feature Layer**: “OBJEKTVAL” = ‘I_Geleis’ AND “TUNNELTYPE” = ‘ ‘

GA G3

- **V25_eis_arc**: **Make Feature Layer**: (“OBJEKTVAL” = ‘Gt_Bahn’ OR “OBJEKTVAL” = NS_BAHN1 OR “OBJEKTVAL” = ‘SS_BAHN1’) AND “TUNNELTYPE” = ‘ ‘

GA G5

- **V25_eis_arc**: **Make Feature Layer**: (“OBJEKTVAL” = ‘NS_BAHN2’ OR “OBJEKTVAL” = ‘SS_BAHN2) AND TUNNELTYPE = ‘ ‘
- **Buffer**: Distance: 10m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **Erase**: Input: DA_Eisenbahn_GA_G1a, Erase: DA_Eisenbahn_GA_G3a
- **Erase (2)**: Input: DA_Eisenbahn_GA_G1a_Erase, Erase: DA_Eisenbahn_GA_G5
- **Erase (3)**: Input: DA_Eisenbahn_GA_G3a, Erase: DA_Eisenbahn_GA_G5
- **Merge**: DA_Eisenbahn_GA_G5, DA_Eisenbahn_GA_G3 und DA_Eisenbahn_GA_G1
- **Dissolve**: Create Multipart Feature: non checked
- **Add Field**: Field Name: DA_Eis, Field Type: SHORT
- **Calculate Field**: Field Name: DA_Eis, Expression: 1

4.2 Anlagen

Modell:	Anlagen GA (Toolbox: SiPro 93 DAMAGE)
Input:	V25_gwn_arc, V25_eob_point, V25_eob_arc, GSM_Sendeanlagen, V25_pri_polygon, V25_geb_polygon und V25_anl_polygon
Output:	DA_Anlage_GA_G1, DA_Anlage_GA_G3, DA_Anlage_GA_G10 und DA_Anlage_swi.

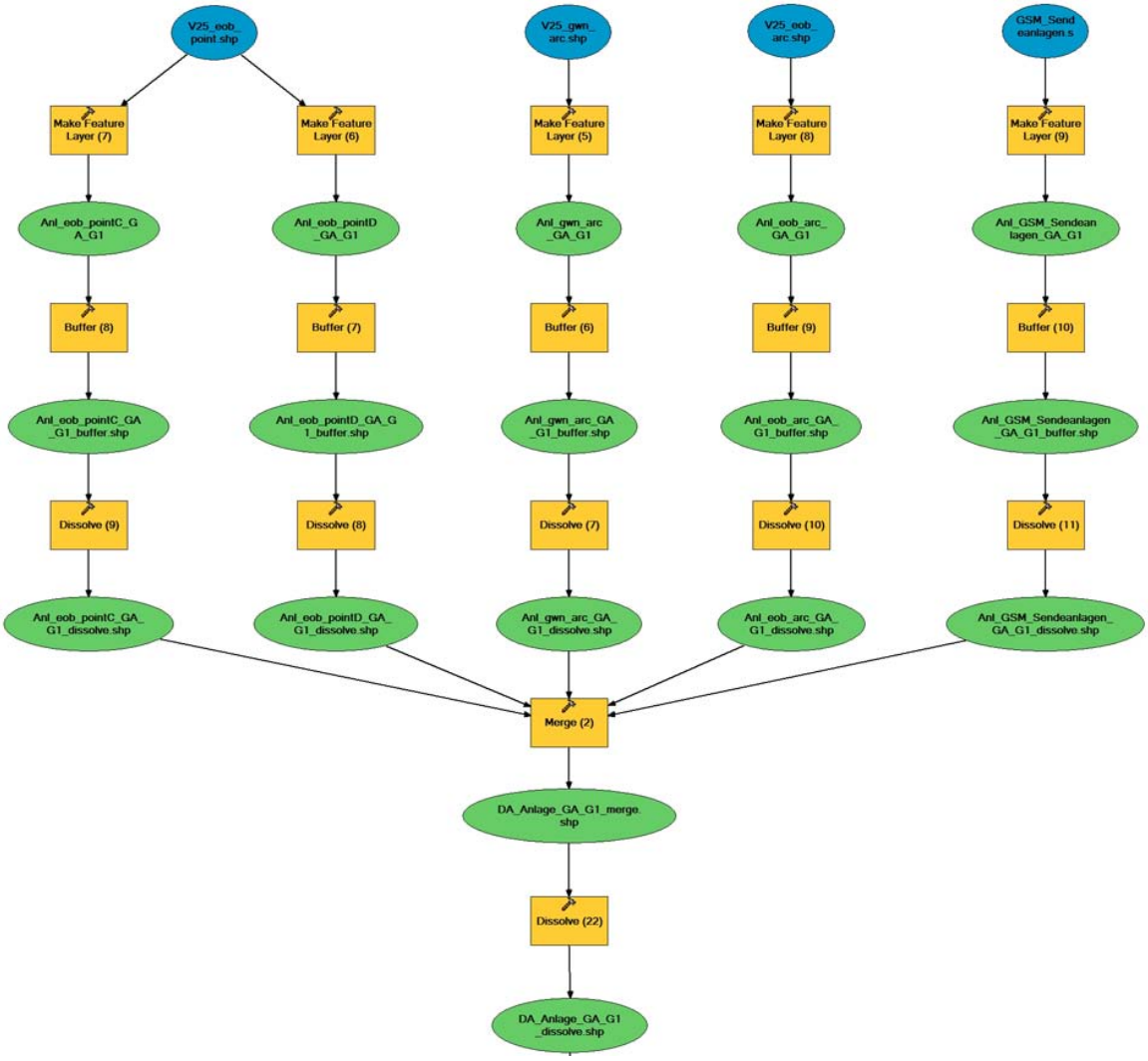
Beschreibung:

Das Modell unterscheidet die Anlagen nach drei Gewichtungen:

- **Gewichtung 1**: Versorgungsstörungen und/oder hohe Sachschäden
    - Objekte: Hochkamin, Reservoir, Antenne, Sendeanlage, Druckleitung einfach, Druckleitung mehrfach, Radiosender und GSM Sendeanlagen

- **Gewichtung 3**: Zeitweiser Aufenthalt bei niedriger Präsenzwahrscheinlichkeit
    - Objekte: Staudamm, Staumauer, Kühlturm, Lagertank, Wasserbecken (Schwimmbäder, ARA), Hafen, Schiffstation, Abwasserreinigungsanlage und Elektrizitätswerk

- **Gewichtung 10**: Dauernde oder beinahe dauernde Präsenz von Menschen
    - Objekte: Bahnhofareal, Flughafenareal und Flughafenbahnhofareal

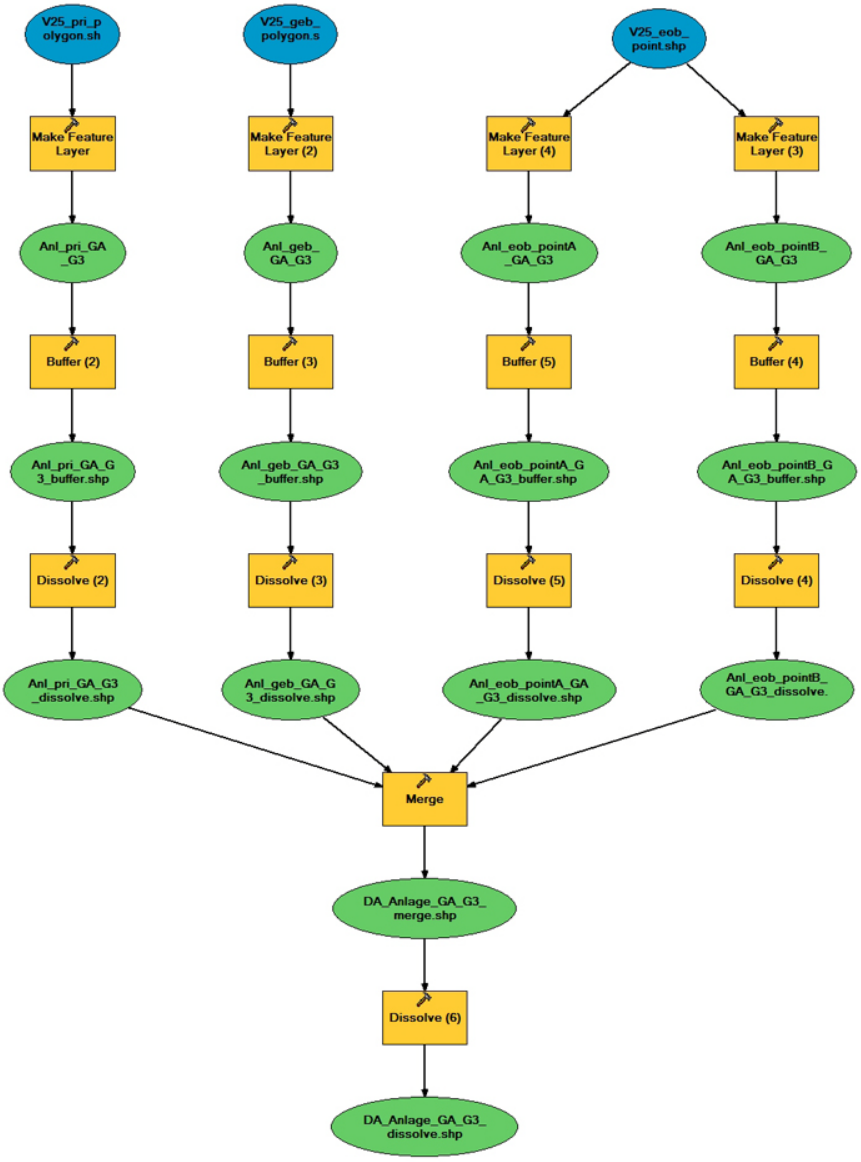


Modell 6: Anlage (Teil 1).

Beschreibung des Modells:

GA G1:

- **V25_eob_point**: **Make Feature Layer**: “OBJEKTVAL“ = ‘Kamin’ OR “OBJEKTVAL“ = ‘Reserv’ OR “OBJEKTVAL“ = ‘W_Turm’
- **Buffer**: Distance: 10m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **V25_eob_point**: **Make Feature Layer**: “OBJEKTVAL“ = ‘Antenne’ OR “OBJEKTVAL“ = ‘SendeAnl’
- **Buffer:** Distance: 5m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **V25_gwn_arc**: **Make Feature Layer**: “OBJEKTVAL“ = ‘Druckl_1’ OR “OBJEKTVAL“ = ‘Druckl_2
- **Buffer**: Distance: 5m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **V25_eob_arc**: **Make Feature Layer**: “OBJEKTVAL“ = ‘Sender’
- **Buffer**: Distance: 10m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **GSM_Sendeanlagen: Make Feature Layer**
- **Buffer**: Distance: 5m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **Merge**: Anl_eob_pointC_GA_G1_dissolve, Anl_eob_pointD_GA_G1_dissolve, Anl_gwn_arc_GA_G1_ dissolve, Anl_eob_arc_GA_G1_dissolve et Anl_GSM_Sendeanlagen_GA_G1_dissolve
- **Dissolve**: Create Multipart Feature: non checked

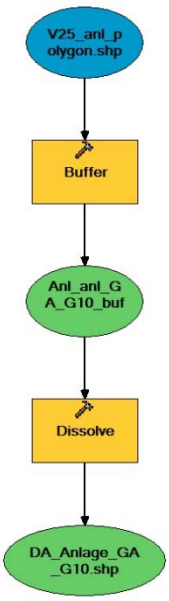


Modell 7: Anlage (Teil 2).

Beschreibung des Modells:

GA G3:

- **V25_pri_polygon**: **Make Feature Layer**: “OBJEKTVAL“ = ‘Z_StauDa’ OR “OBJEKTVAL“ = ‘Z_StauMa’
- **Buffer**: Distance: 25m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **V25_geb_polygon**: **Make Feature Layer**: “OBJEKTVAL“ = ‘Z_Kuehlturm’ OR “OBJEKTVAL“ = ‘Z_Lagertank’ OR “OBJEKTVAL“ = ‘Z_WBecken’
- **Buffer**: Distance: 25m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **V25_eob_pointA**: **Make Feature Layer**: “OBJEKTVAL“ = ‘Hafen’ OR “OBJEKTVAL“ = ‘Schiffst’
- **Buffer**: Distance: 25m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **V25_eob_pointB**: **Make Feature Layer**: “OBJEKTVAL“ = ‘ARA’ OR “OBJEKTVAL“ = ‘ElWerk’
- **Buffer**: Distance: 10m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **Merge**: Anl_eob_pointB_GA_G3_dissolve, Anl_eob_pointA_GA_G3_dissolve, Anl_pri_GA_G3_dissolve et Anl_geb_GA_G3_dissolve
- **Dissolve**: Create Multipart Feature: non checked

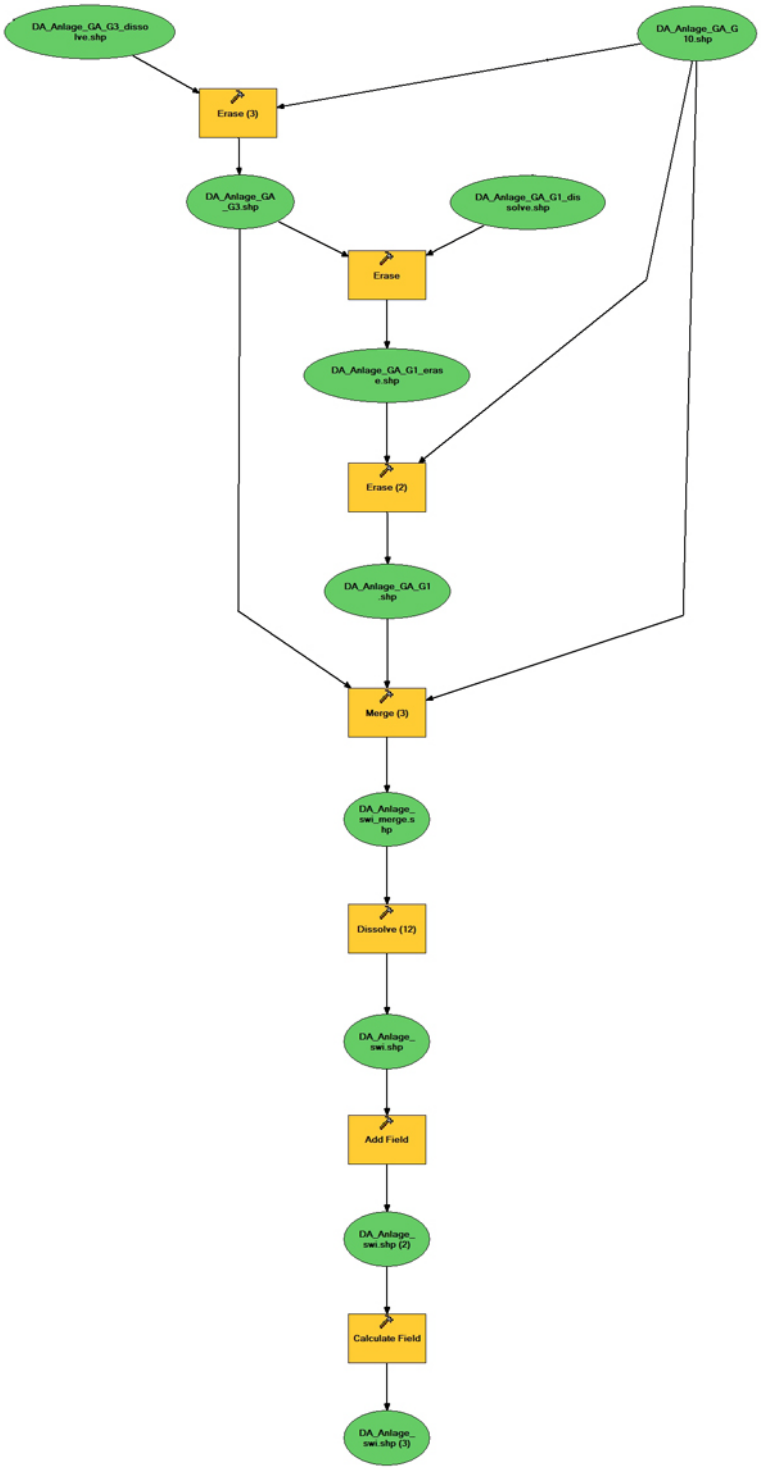


Modell 8: Anlage (Teil 3).

Beschreibung des Modells:

GA G10:

- **V25_anl_polygon**: **Buffer**: Distance: 25m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked



Modell 9: Anlage (Teil 4).

Beschreibung des Modells:

- **Erase (3):** Input Features: DA_Anlage_GA_G3_dissolve, Erase Features: DA_Anlage_GA_G10
- **Erase:** Input Features: DA_Anlage_GA_G1_dissolve, Erase DA_Anlage_Ga_G3
- **Erase (2):** Input Features: DA_anlage_GA_G1_erase, Erase Features: DA_Anlage_GA_G10
- **Merge**: DA_Anlage_GA_G10, DA_Anlage_GA_G3 und DA_Anlage_GA_G1
- **Dissolve**: Create Multipart Feature: non checked
- **Add Field**: Field Name: DA_Anl, Field Type: SHORT
- **Calculate Field**: Field Name: DA_Anl, Expression: 1

4.3 Gebäude

Bei den Gebäuden werden vier verschiedenen Kategorien unterschieden:

- Wohngebäude ständig bewohnt (Kapitel 4.3.1)
- Wohngebäude zeitweise bewohnt (Kapitel 4.3.2)
- Industriegebäude (Kapitel 4.3.3) geteilt in Unterkategorien:
    - Industrie Punkt (Kapitel 4.3.3.1),
      Aus der Betriebszählung wurden all jene Objekte ausgewählt, die nicht auf einem Gebäude vom Datensatz „Vector25 Gebäude“ liegen und nicht zu den Kategorien „Anlagen“ oder „öffentliche Gebäuden“ gehören.
    - Industrie Klein (Kapitel 4.3.3.2),
      Aus der Betriebszählung wurden all jene Objekte ausgewählt, die auf einem Gebäude vom Datensatz „Vector25 Gebäude“ liegen und nicht zu den Kategorien „Anlagen“ oder „öffentliche Gebäuden“ gehören. War die Fläche des Punktepuffers von 25m grösser als die Fläche des Gebäudes, wurde der gepufferte Punkt verwendet.
    - Industrie Gross (Kapitel 4.3.3.3),
      Um dem Umstand Rechnung zu tragen, dass grössere Industrieareale in der Betriebszählung nur durch einen Punkt repräsentiert sind und somit durch reine Pufferung massiv unterschätzt würden, wurde im Falle, dass die Gebäudefläche grösser als die Pufferfläche ist, die Gebäudefläche aus dem Datensatz „Vector25 Gebäude“ verwendet. Dadurch konnten grössere Industrieareale realistisch abgebildet werden.
    - Industrie Areal (Kapitel 4.3.3.4),
      Ein mehrere Gebäude umfassendes Industrieareal wird in der Betriebszählung nur durch einen Koordinatenpunkt repräsentiert. Aus diesem Grund wurden zusätzlich alle Gebäude aus dem Datensatz „Vector25 Gebäude“ ausgewählt, die in der Industriezone aus der Bodennutzung 1997 liegen (und nicht zu den Kategorien „Anlagen“ oder „öffentliche Gebäuden“ gehören).
- Öffentliche Gebäude (Kapitel 4.3.4)

Beschreibung:

Die Modelle unterscheiden die Gebäude nach drei Gewichtungen:

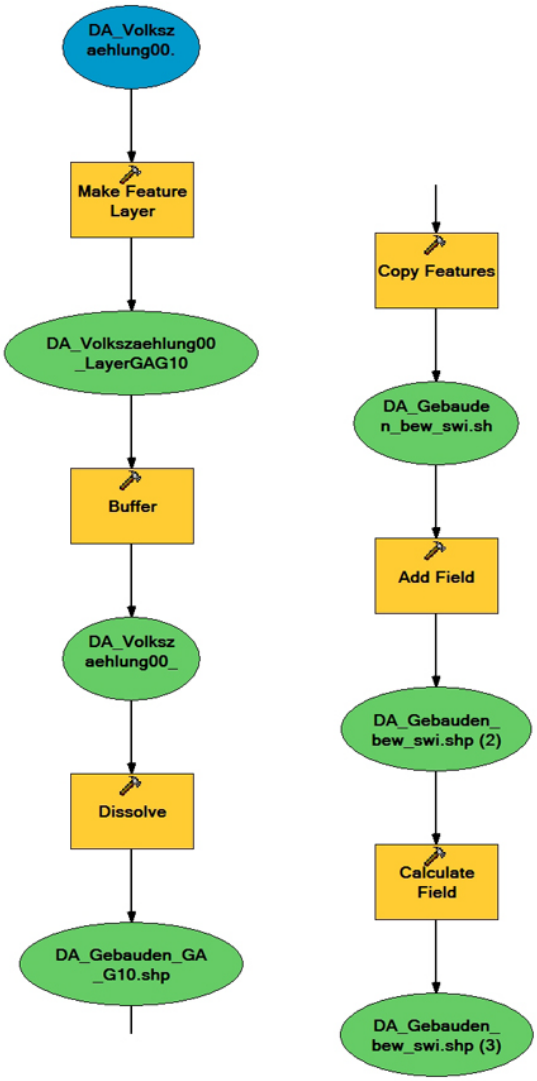
- **Gewichtung 3**: Zeitweiser Aufenthalt bei niedriger Präsenzwahrscheinlichkeit
    - Objekte: Öffentliche Gebäude wie Kirche, Perrondach, Schloss / Burg und Station / ÖV Haltestelle

- **Gewichtung 5**: Zeitweiser Aufenthalt bei hoher Präsenzwahrscheinlichkeit
    - Objekte: Wohngebäude zeitw. bewohnt, Industrie- und Gewerbegebäude

- **Gewichtung 10**: Dauernde oder beinahe dauernde Präsenz von Menschen
    - Objekte: Wohngebäude ständig bewohnt

4.3.1 Wohngebäude ständig bewohnt

Modell: Gebäude GA10 (Toolbox: SiPro 93 DAMAGE)
Input: DA_Volkszaehlung00
Output: DA_Gebauden_GA_G10 und DA_Gebauden_bew_swi.



Modell 10: Wohngebäude ständig bewohnt.

Beschreibung des Modells:

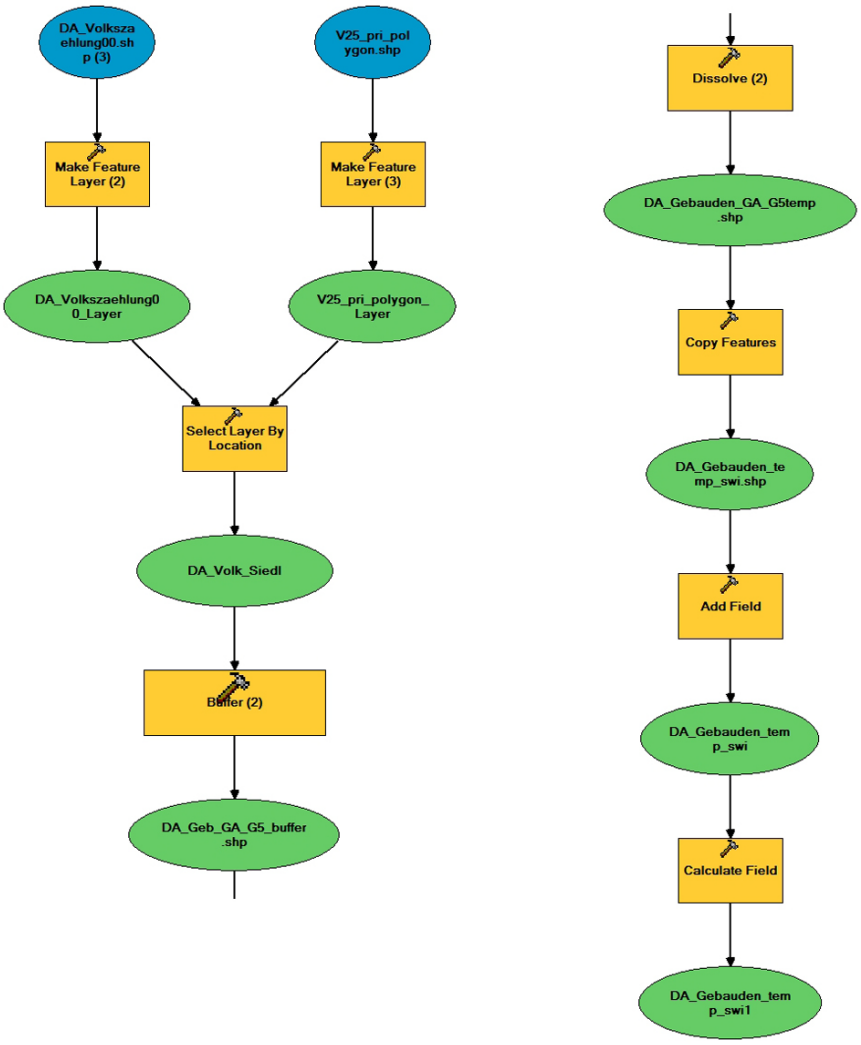
- **DA_Volkszaehlung00**: **Make Feature Layer**: A00WDTOT > 0
- **Buffer**: Distance: 25m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **Copy Feature**: DA_Gebauden_GA_G10
- **Add Field**: Field Name: DA_Geb_bew, Field Type: SHORT
- **Calculate Field**: Field Name: DA_Geb_bew, Expression: 1

4.3.2 Wohngebäude zeitweise bewohnt

Modell: Gebäude GA5 temp (Toolbox: SiPro 93 DAMAGE)

Input: DA_Volkszaehlung00 und V25_pri_polygon

Output: DA_Gebauden_GA_G5temp und DA_Gebauden_temp_swi.



Modell 11: Wohngebäude zeitweise bewohnt.

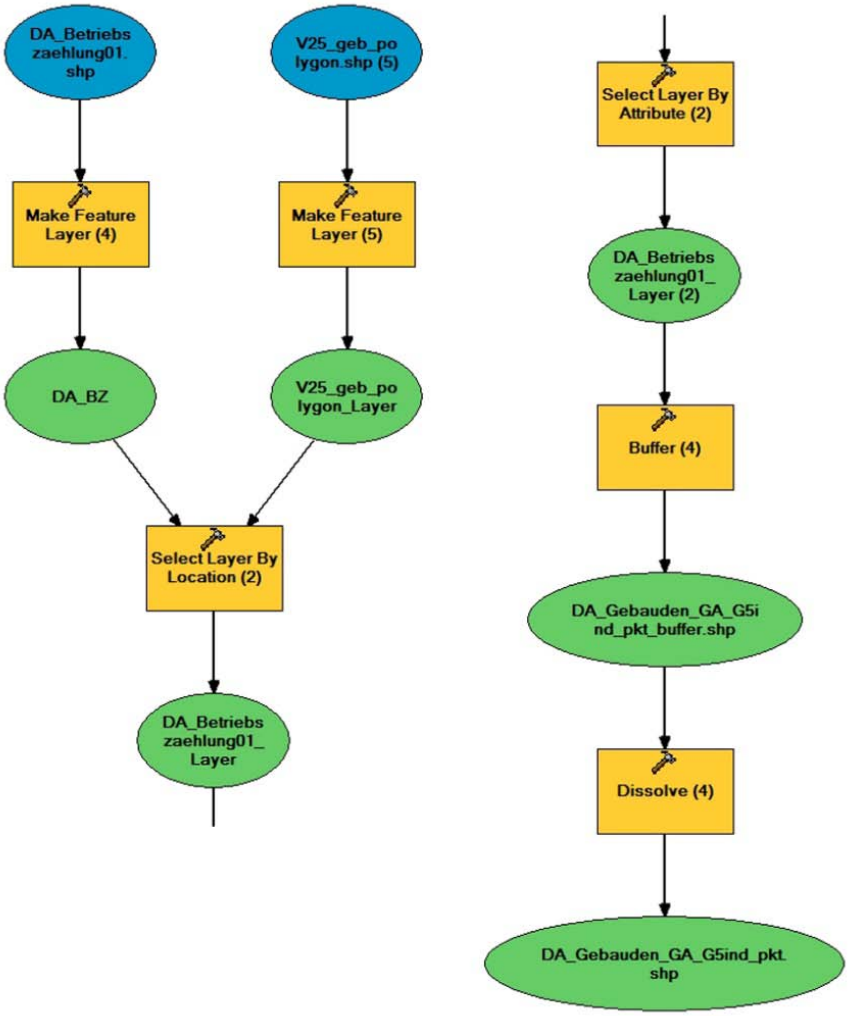
Beschreibung des Modells:

- **V25_pri_polygon**: **Make Feature Layer**: “OBJEKTVAL = ‘Z_Siedl’
- **DA_Volkzaehlung00: Make Feature Layer**: “A00WDTOT “ 0 AND “A00WTZTOT” > 0
- **Select Layer By Location**: Input: DA_Volkzaehlung_Layer, Selecting: V25_pri_polygon_Layer
- **Buffer**: Distance: 25m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **Copy Feature**: DA_Gebauden_GA_G5temp
- **Add Field**: Field Name: DA_Geb_bew, Field Type: SHORT
- **Calculate Field**: Field Name: DA_Geb_bew, Expression: 1

4.3.3 Industriegebäude

4.3.3.1 Industrie Punkt

Modell:	Gebäude GA5 ind punkt (Toolbox: SiPro 93 DAMAGE)
Input:	DA_Betriebezaehlung und V25_geb_polygon
Output:	DA_Gebauden_GA_G5ind_pkt.



Modell 12: Industrie Punkt.

Beschreibung des Modells:

- **DA_Betriebezaehlung01: Make Feature Layer**
- **V25_geb_polygon**: **Make Feature Layer**: "OBJEKTVAL = 'Z_Gasthof' OR "OBJEKTVAL = 'Z_Gebaeude' OR "OBJEKTVAL = 'Z_Huette' OR "OBJEKTVAL = 'Z_Schiessstand' OR "OBJEKTVAL = 'Z_Treibhaus' OR "OBJEKTVAL = 'Z_Innenhof'
- **Select Layer By Location**: Input: DA_BZ, Selecting: V25_geb_polygon_Layer
- **Select Layer By Attribute**: Layer Name: DA_Betriebszaehlung01_Layer, Selection type: SWITCH SELECTION
- **Buffer**: Distance: 25m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked

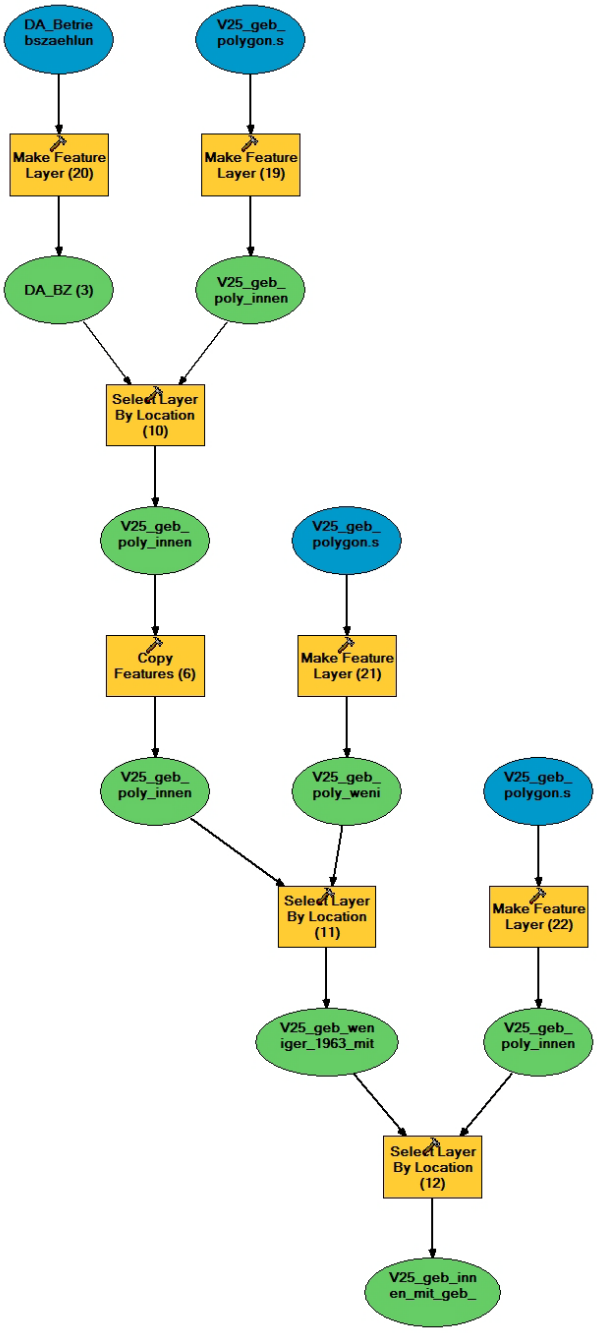
4.3.3.2 Industrie Klein

Modell:	Gebäude GA5 ind klein (Toolbox: SiPro 93 DAMAGE)
Input:	DA_Betriebezaehlung und V25_geb_polygon
Output:	DA_Gebauden_GA_G5ind_ klein.

Bemerkung zur Schwierigkeit bezüglich Innenhöfe:

Manchmal sind die Punkte (X und Y Koordinaten) aus der Betriebzählung nicht auf einem Gebäude sondern im Innenhof des Gebäudes abgelegt. Um dieses Problem zu beheben, mussten erst die Innenhöfe selektiert und dann die betroffenen Gebäude davon abgeleitet werden. Diese Arbeitsschritte sind direkt im Modell integriert.

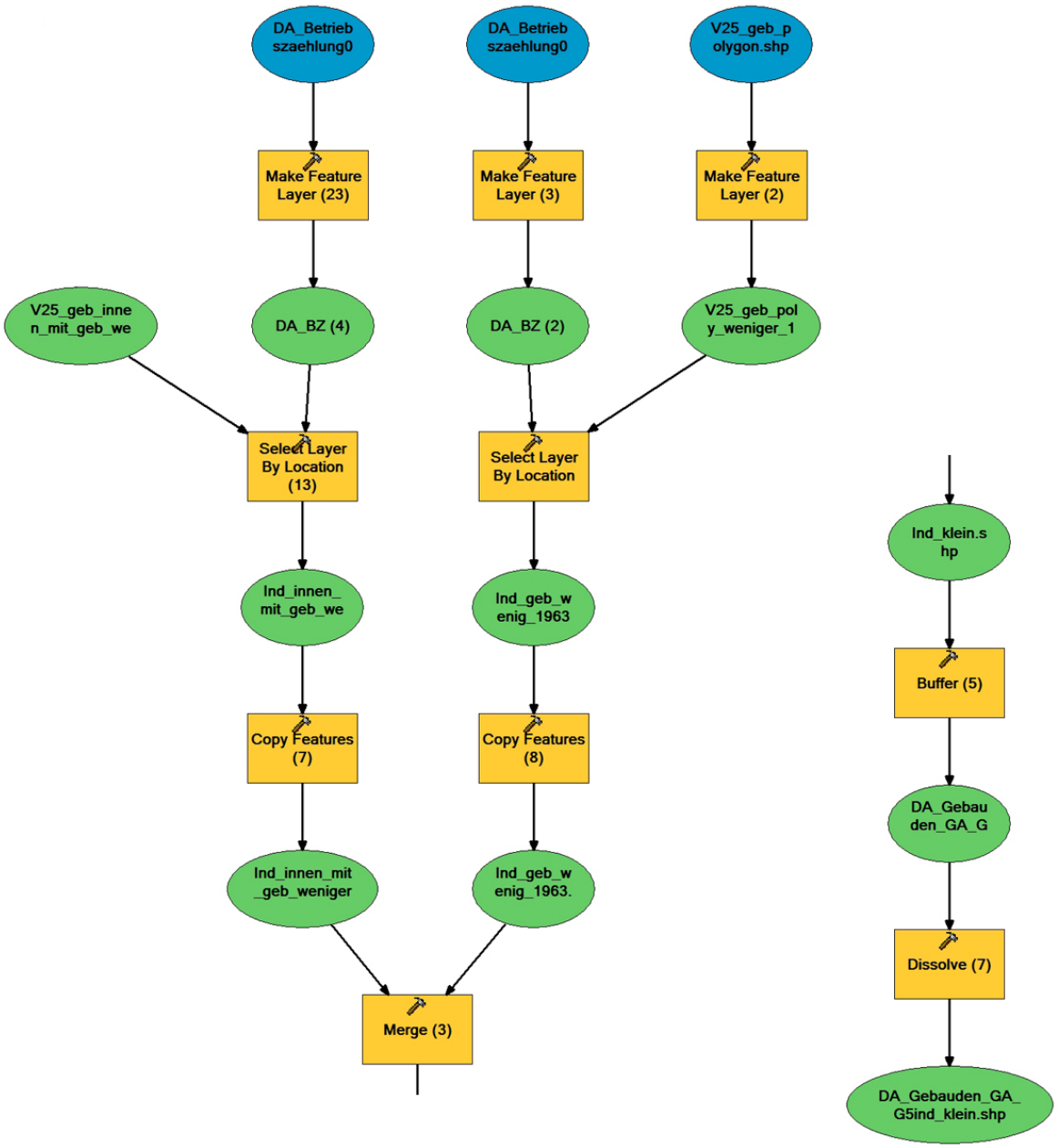
Abbildung 24: Koordinaten der Betriebszählung im Innenhof eines Gebäudes.



Modell 13: Industrie Klein abgeleitet von den Daten aus der Betriebszählung (Gebäudeinnenhöfe).

Beschreibung des Modells:

- **DA_Betriebezaehlung01: Make Feature Layer**
- **V25_geb_polygon**: **Make Feature Layer**: "OBJEKTVAL" = 'Z_Innenhof'
- **Select Layer By Location**: Input: V25_geb_poly_innen, Selecting: DA_BZ (3)
- **Copy Feature**: Input: V25_geb_poly_innen_mit_BZ
- **V25_geb_polygon**: **Make Feature Layer**: ("OBJEKTVAL" = 'Z_Gasthof' OR "OBJEKTVAL" = 'Z_Gebaeude' OR "OBJEKTVAL" = 'Z_Huette' OR "OBJEKTVAL" = 'Z_Schiessstand' OR "OBJEKTVAL" = 'Z_Treibhaus') AND "AREA" <= 1963,5
- **Select Layer By Location**: Input: V25_geb_poly_weniger_1963 (2), Selecting: V25_geb_poly_innen_mit_BZ
- **V25_geb_polygon**: **Make Feature Layer**: "OBJEKTVAL" = 'Z_Innenhof'
- **Select Layer By Location**: Input: V25_geb_poly_innen (2), Selecting: V25_geb_weniger_1963_mit_innen



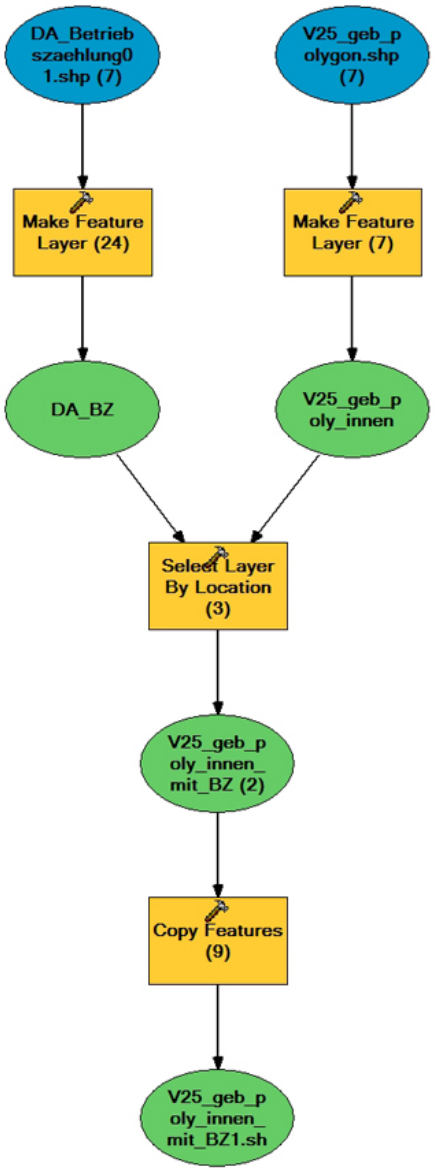
Modell 14: Industrie Klein (Fortsetzung).

Beschreibung des Modells:

- **DA_Betriebezaehlung01: Make Feature Layer**
- **Select Layer By Location**: Input: DA_BZ (4), Selecting: V25_geb_innen_mit_geb_weniger_1963
- **Copy Feature**: Input: Ind_innen_mit_geb_weniger_1963
- **DA_Betriebezaehlung01: Make Feature Layer**
- **V25_geb_polygon**: **Make Feature Layer**: ("OBJEKTVAL" = 'Z_Gasthof' OR "OBJEKTVAL" = 'Z_Gebaeude' OR "OBJEKTVAL" = 'Z_Huette' OR "OBJEKTVAL" = 'Z_Schiessstand' OR "OBJEKTVAL" = 'Z_Treibhaus') AND "AREA" <= 1963,5
- **Select Layer By Location**: Input: DA_BZ (2), Selecting: V25_geb_poly_weniger_1963
- **Copy Feature**: Input: Ind_geb_wenig_1963
- **Merge**: Input: Ind_innen_mit_geb_weniger_1963.shp und Ind_geb_wenig_1963
- **Buffer**: Distance: 25m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked

4.3.3.3 Industrie Gross

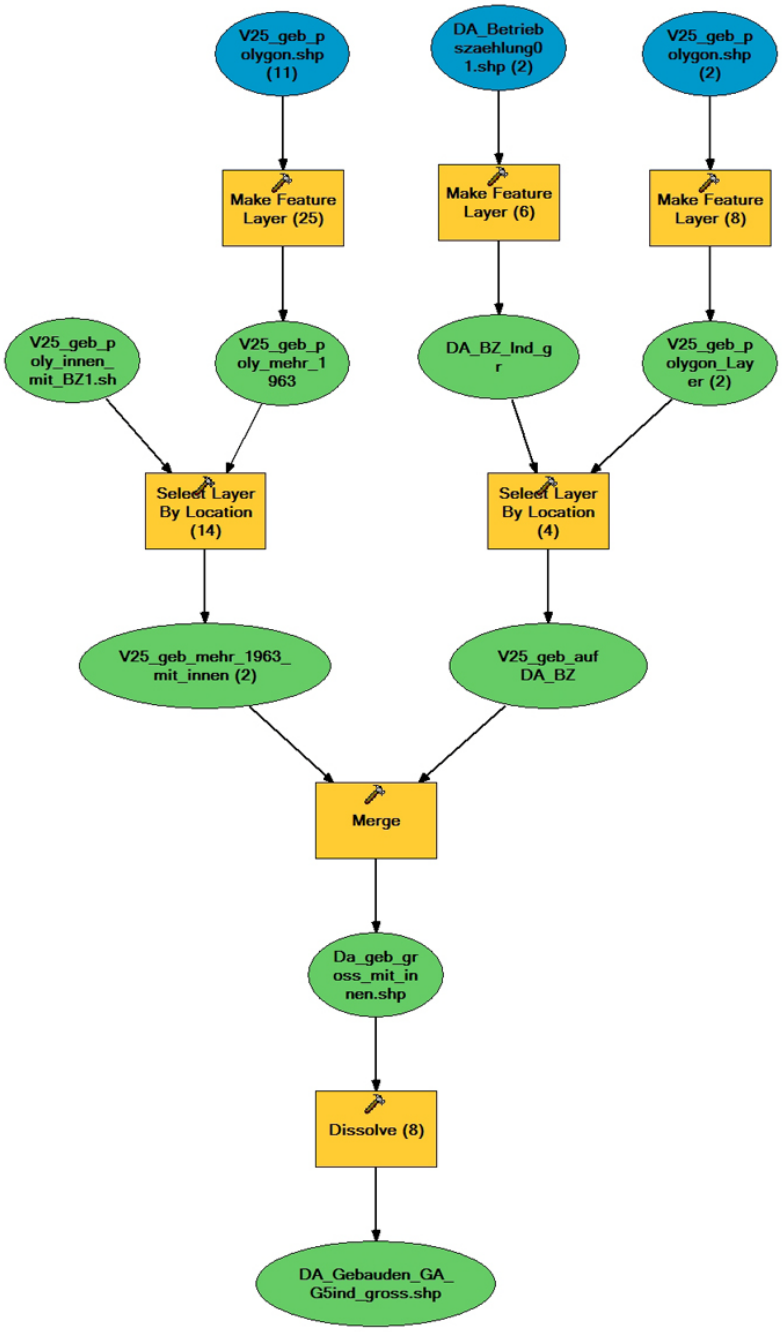
Modell:	Gebäude GA5 ind gross (Toolbox: SiPro 93 DAMAGE)
Input:	DA_Betriebezaehlung und V25_geb_polygon
Output:	DA_Gebauden_GA_G5ind_ gross.



Modell 15: Industrie Gross abgeleitet von den Daten aus der Betriebszählung (Gebäudeinnenhöfe).

Beschreibung des Modells:

- **DA_Betriebezaehlung01: Make Feature Layer**
- **V25_geb_polygon**: **Make Feature Layer**: "OBJEKTVAL" = 'Z_Innenhof'
- **Select Layer By Location**: Input: V25_geb_poly_innen, Selecting: DA_BZ
- **Copy Feature**: Input: V25_geb_poly_innen_mit_BZ



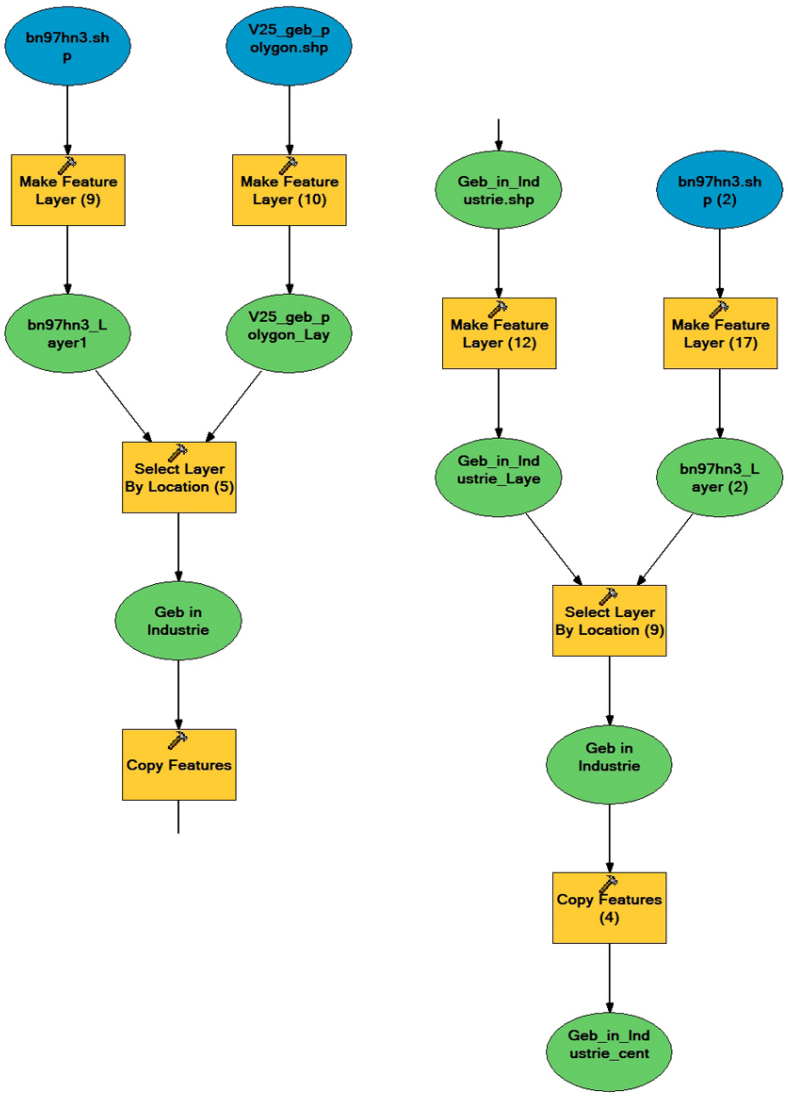
Modell 16: Industrie Gross (Fortsetzung).

Beschreibung des Modells:

- **V25_geb_polygon**: **Make Feature Layer**: ("OBJEKTVAL" = 'Z_Gasthof' OR "OBJEKTVAL" = 'Z_Gebaeude' OR "OBJEKTVAL" = 'Z_Huette' OR "OBJEKTVAL" = 'Z_Schiessstand' OR "OBJEKTVAL" = 'Z_Treibhaus') AND "AREA" > 1963,5
- **Select Layer By Location**: Input: V25_geb_poly_mehr_1963, Selecting: V25_geb_poly_innen_mit_BZ1
- **DA_Betriebezaehlung01: Make Feature Layer**
- **V25_geb_polygon**: **Make Feature Layer**: ("OBJEKTVAL" = 'Z_Gasthof' OR "OBJEKTVAL" = 'Z_Gebaeude' OR "OBJEKTVAL" = 'Z_Huette' OR "OBJEKTVAL" = 'Z_Schiessstand' OR "OBJEKTVAL" = 'Z_Treibhaus') AND "AREA" > 1963,5
- **Select Layer By Location**: Input: V25_geb_polygon_Layer (2), Selecting: DA_BZ_Ind_gr
- **Merge**: Input: V25_geb_auf DA_BZ und V25_geb_mehr_1963_mit_innen (2)
- **Dissolve**: Create Multipart Feature: non checked

4.3.3.4 Industrie Areal

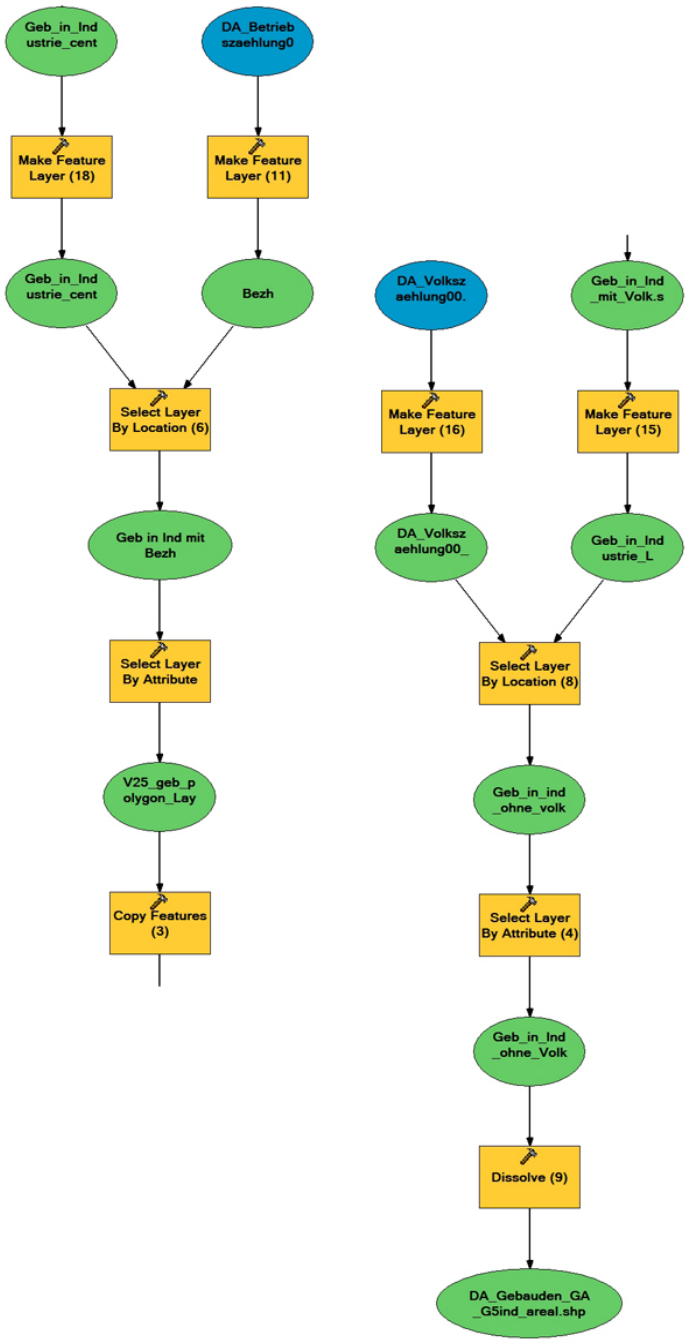
Modell:	Gebäude GA5 ind areal (Toolbox: SiPro 93 DAMAGE)
Input:	bn97hn3, DA_Betriebezaehlung, DA_Volkzaehlung00 und V25_geb_polygon
Output:	DA_Gebauden_GA_G5ind_ areal.



Modell 17: Industrie Areal (Teil 1).

Beschreibung des Modells:

- **Bn97hn3**: **Make Feature Layer**: "GRIDCODE" = 15
- **V25_geb_polygon**: Make Feature Layer: "OBJEKTVAL = 'Z_Gasthof' OR "OBJEKTVAL = 'Z_Gebaeude' OR "OBJEKTVAL = 'Z_Huette' OR "OBJEKTVAL = 'Z_Schiessstand' OR "OBJEKT-VAL = 'Z_Treibhaus'
- **Select Layer By Location**: Input: V25_geb_polygon_Layer, Selecting: bn97hn3_Layer1
- **Copy Feature**: Input: Geb_in_Industrie
- **Geb_in_Industrie**: **Make Feature Layer**
- **Bn97hn3: Make Feature Layer**: "GRIDCODE" = 15
- **Select Layer By Location**: Input: Geb_in_Industrie_Layer, Relationship: HAVE_THEIR_CENTRE_IN Selecting: bn97hn3_Layer (2)
- **Copy Feature**: Input: Geb in Industrie (2)



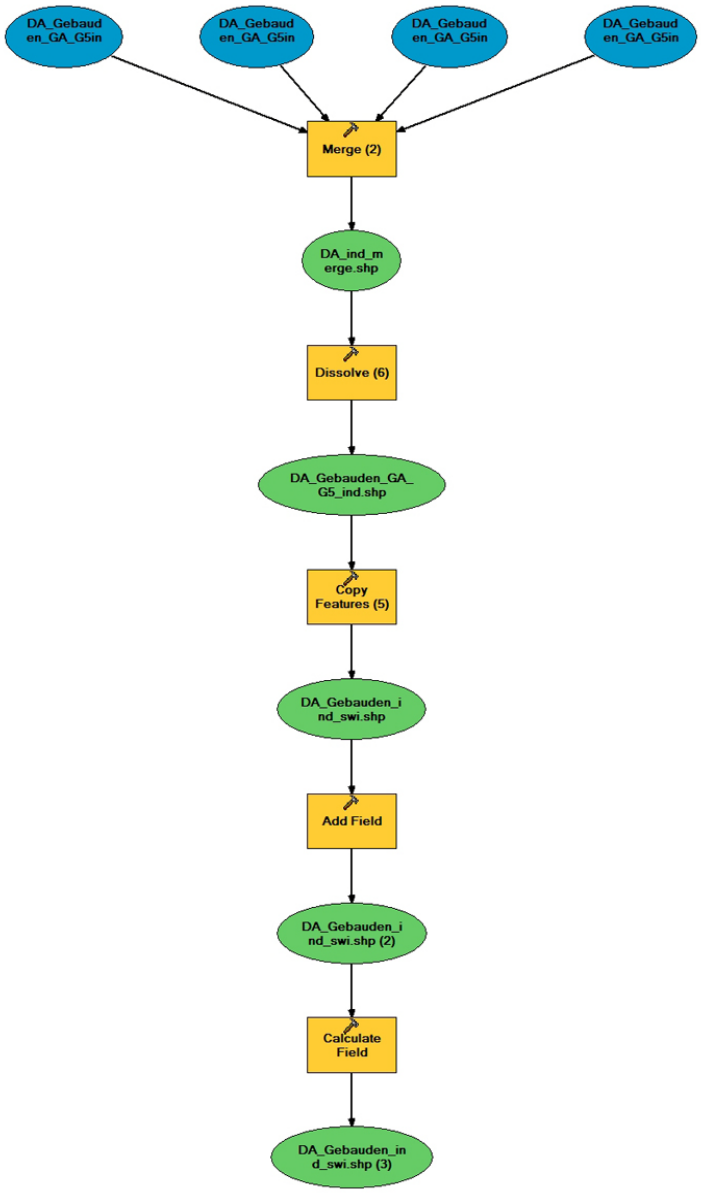
Modell 18: Industrie Areal (Teil 2).

Beschreibung des Modells:

- **Geb_in_Industrie_center**: **Make Feature Layer**
- **DA_Betriebszaehlung01: Make Feature Layer**
- **Select Layer By Location**: Input: Geb_in_Industrie_center_Layer, Selecting: Bezh
- **Select Layer By Attribute**: Layer Name: Geb_in_Ind_mit_Bezh, Selection type: SWITCH_SELECTION
- **Copy Feature**: V25_geb_polygon_Layer2 (2)
- **DA_Volkszaehlung00_Layer: Make Feature Layer**
- **Geb_in_Ind_mit_Volk**: **Make Feature Layer**
- **Select Layer By Location**: Input: Geb_in_Industrie_L, Selecting: DA_VolksZaehlung00_Layer1
- **Select Layer By Attribute**: Layer Name: Geb_in_Ind_ohne_Volk, Selection type: SWITCH_SELECTION
- **Dissolve**: Create Multipart Feature: non checked

4.3.3.5 Industrie Total

Modell:	Gebäude GA5 ind (Toolbox: SiPro 93 DAMAGE)
Input:	DA_gebauden_GA5 ind pkt, DA_gebauden_GA5 ind klein, DA_gebauden_GA5 ind_gross und DA_gebauden_GA5 ind areal
Output:	DA_Gebauden_GA_G5ind und DA_Gebauden_ind_swi.



Modell 19: Gruppierung der Industrien mit der Gewichtung 5.

Beschreibung des Modells:

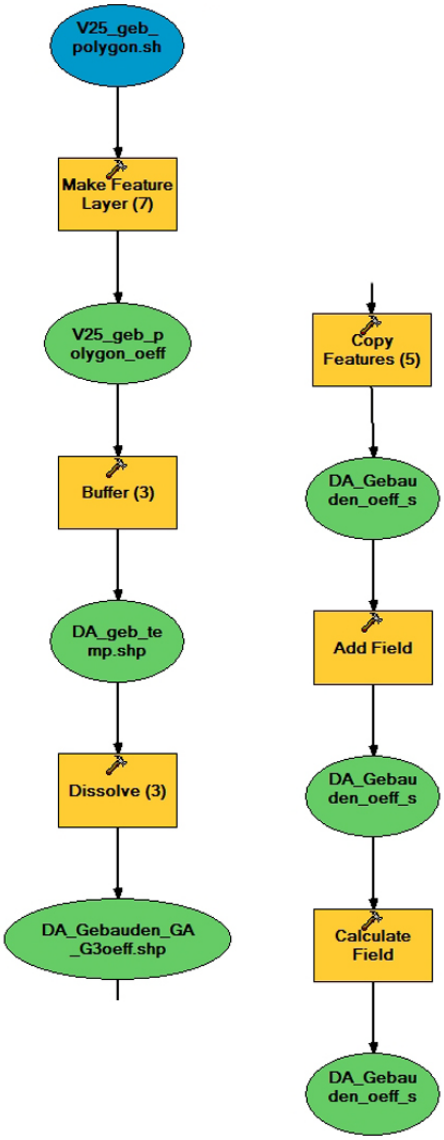
- **Merge**: Input: DA_gebauden_GA5 ind pkt, DA_gebauden_GA5 ind klein, DA_gebauden_GA5 ind gross und DA_gebauden_GA5 ind areal
- **Dissolve**: Create Multipart Feature: non checked
- **Copy Feature**: Input: DA_Gebauden_GA_G5_ind
- **Add Field**: Field Name: DA_ind, Field Type: SHORT
- **Calculate Field**: Field Name: DA_ind, Expression: 1

4.3.4 Öffentliche Gebäude

Modell: Gebäude GA3 oeff (Toolbox: SiPro 93 DAMAGE)

Input: V25_geb_polygon

Output: DA_Gebauden_GA_G3oeff und DA_Gebauden_oeff_swi.



Modell 20: Öffentliche Gebäude.

Beschreibung des Modells:

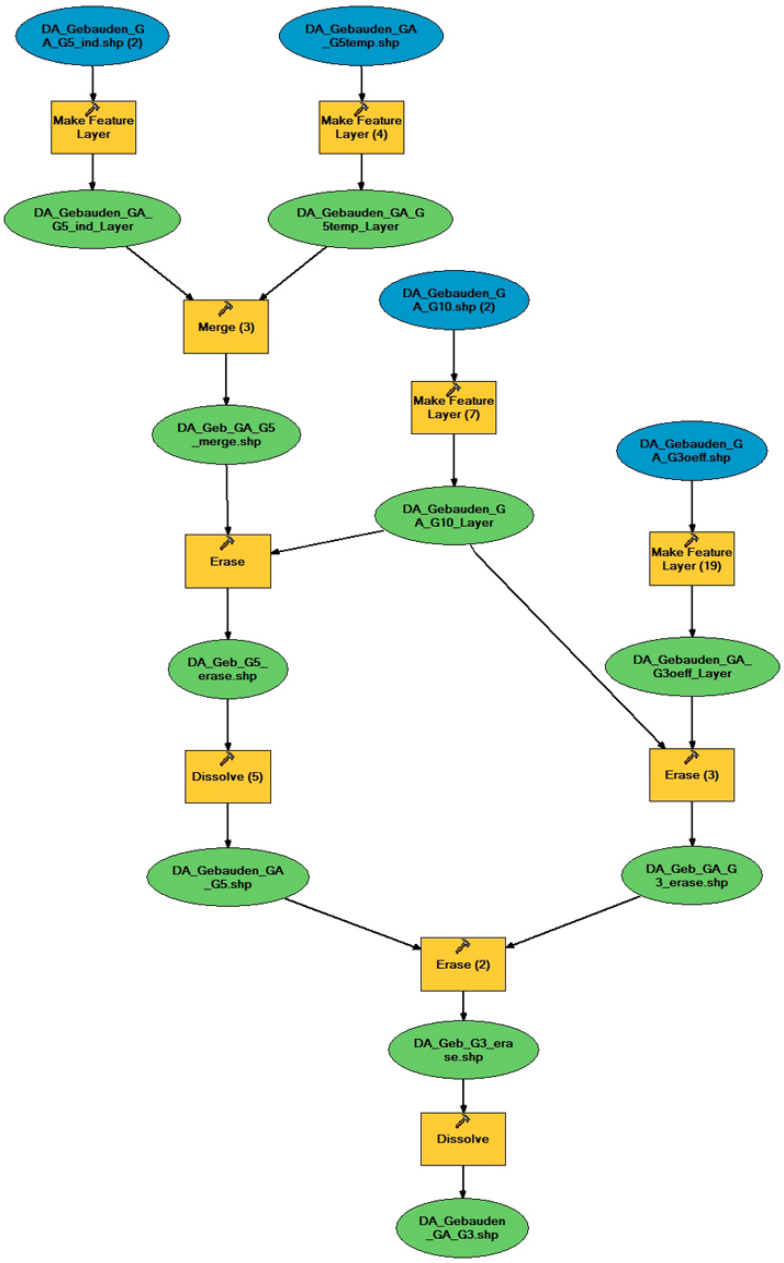
- **V25_geb_polygon**: **Make Feature Layer**: "OBJEKTVAL = 'Z_Kirche' OR "OBJEKTVAL = 'Z_Perron' OR "OBJEKTVAL = 'Z_Schloss' OR "OBJEKTVAL = 'Z_Station'
- **Buffer**: Distance: 25m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **Copy Feature**: DA_Gebauden_GA_G5oeff
- **Add Field**: Field Name: DA_Geb_bew, Field Type: SHORT
- **Calculate Field**: Field Name: DA_Geb_bew, Expression: 1

4.3.5 Gruppierung Gebäude mit Gewichtung 3 und 5

Modell: Gebäude GA5 GA3 (Toolbox: SiPro 93 DAMAGE)

Input: DA_Gebauden_GA_G5ind, DA_Gebauden_GA_G5temp, DA_Gebauden_GA_G3_oeff und DA_Gebauden_GA_G10.

Output: DA_Gebauden_GA_G5 und DA_Gebauden_GA_G3.



Modell 21: Gebäude mit den Gewichtungen 3 und 5.

Beschreibung des Modells:

- **Erase**: Input: DA_GEB_GA_G5_merge, Erase: DA_Gebauden_GA_G10_Layer
- **Dissolve**: Create Multipart Feature: non checked
- **Erase (3)**: Input: DA_Gebauden_GA_G5oeff_Layer, Erase: DA_Gebauden_GA_G10_Layer
- **Dissolve**: Create Multipart Feature: non checked
- **Erase (2)**: Input: DA_Geb_GA_G3_erase, Erase: DA_Gebauden_GA_G5_Layer
- **Dissolve**: Create Multipart Feature: non checked

4.4 Strassennetz

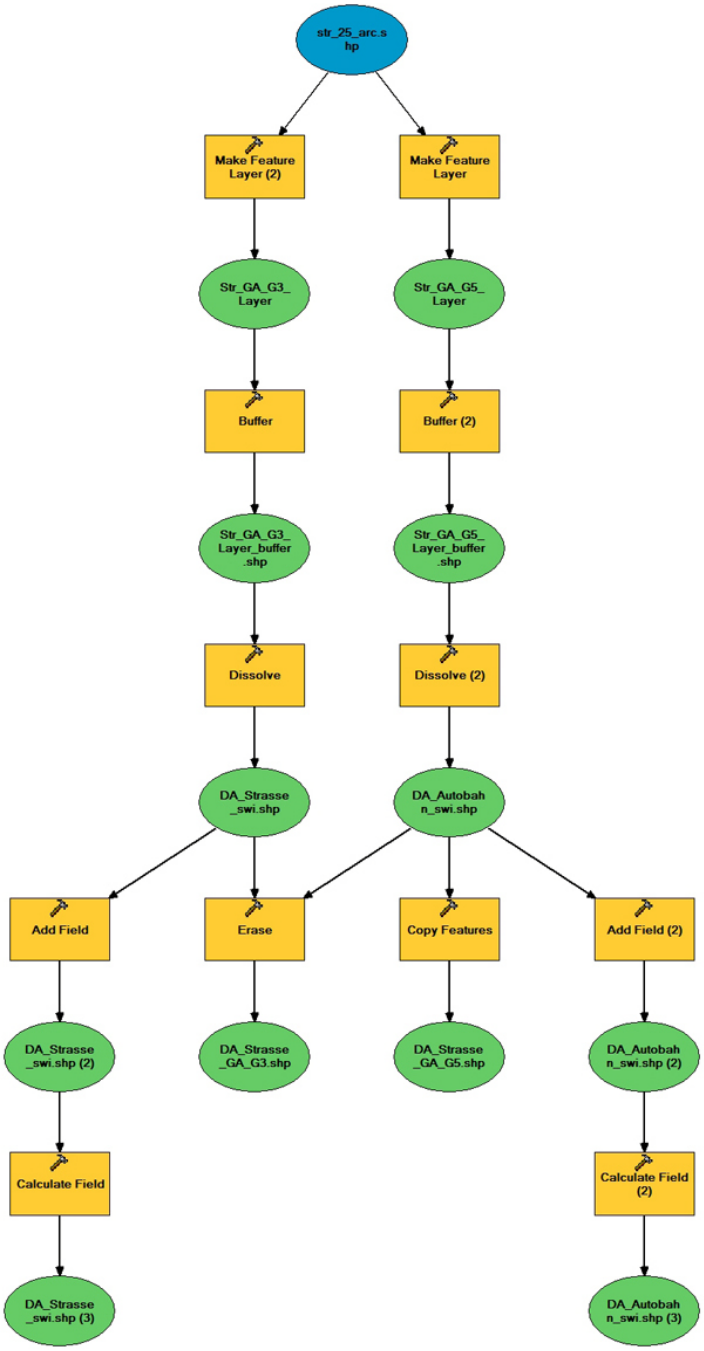
Modell: Strassen GA (Toolbox: SiPro 93 DAMAGE)
Input: Str_25_arc.
Output: DA_Strasse_GA_G5, DA_Strasse_GA_G3, DA_Strassen_swi und DA_Autobahn_swi.

Beschreibung des Modells:

Das Modell unterscheidet das Strassennetz nach zwei Gewichtungen:

- **Gewichtung 3**: Zeitweiser Aufenthalt bei niedriger Präsenzwahrscheinlichkeit
  - Objekte: Strassen 1. und 2. Klasse
  - Bemerkung: kein Tunnel

- **Gewichtung 5**: Zeitweiser Aufenthalt bei hoher Präsenzwahrscheinlichkeit
  - Objekte: Autobahn, Autobahn richtungsgetrennt, Autostrasse, Ein- / Ausfahrt (Autobahn / -strasse) und Autobahnzufahrt
  - Bemerkung: kein Tunnel



Modell 22: Strassennetz.

Beschreibung des Modells:

- **Str_25_arc**: **Make Feature Layer**: ("OBJECTVAL" = 'Autobahn' OR "OBJECTVAL" = 'Autob_Ri' OR "OBJECTVAL" = 'Autostr' OR "OBJECTVAL" = 'Ein_Ausf' OR "OBJECTVAL" = 'A_Zufahrt') AND "TUNNELTYPE" = " "
- **Buffer**: Distance: 10m, End Type: ROUND
- **Dissolve**: Create Multipart Feature: non checked
- **Add Field**: Field Name: DA_Str, Field Type: SHORT
- **Calculate Field**: Field Name: DA_Str, Expression: 1
- **Str_25_arc**: **Make Feature Layer** (2): ("OBJECTVAL" = '1_Klass AND "TUNNELTYPE" = " "
- **Buffer (2)**: Distance: 10m, End Type: ROUND
- **Dissolve (2)**: Create Multipart Feature: non checked
- **Add Field (2)**: Field Name: DA_Autob, Field Type: SHORT
- **Calculate Field (2)**: Field Name: DA_Autob, Expression: 1
- **Erase**: Input Features: DA_Strassen_swi, Erase Features: DA_Autobahn_swi
- **Copy Feature**: Input Features: DA_Autobahn_swi

4.5 Zusammenfassung des Schadenpotenzials

4.5.1 Schadenpotenzial für die Berechnung des Schutzwaldindex

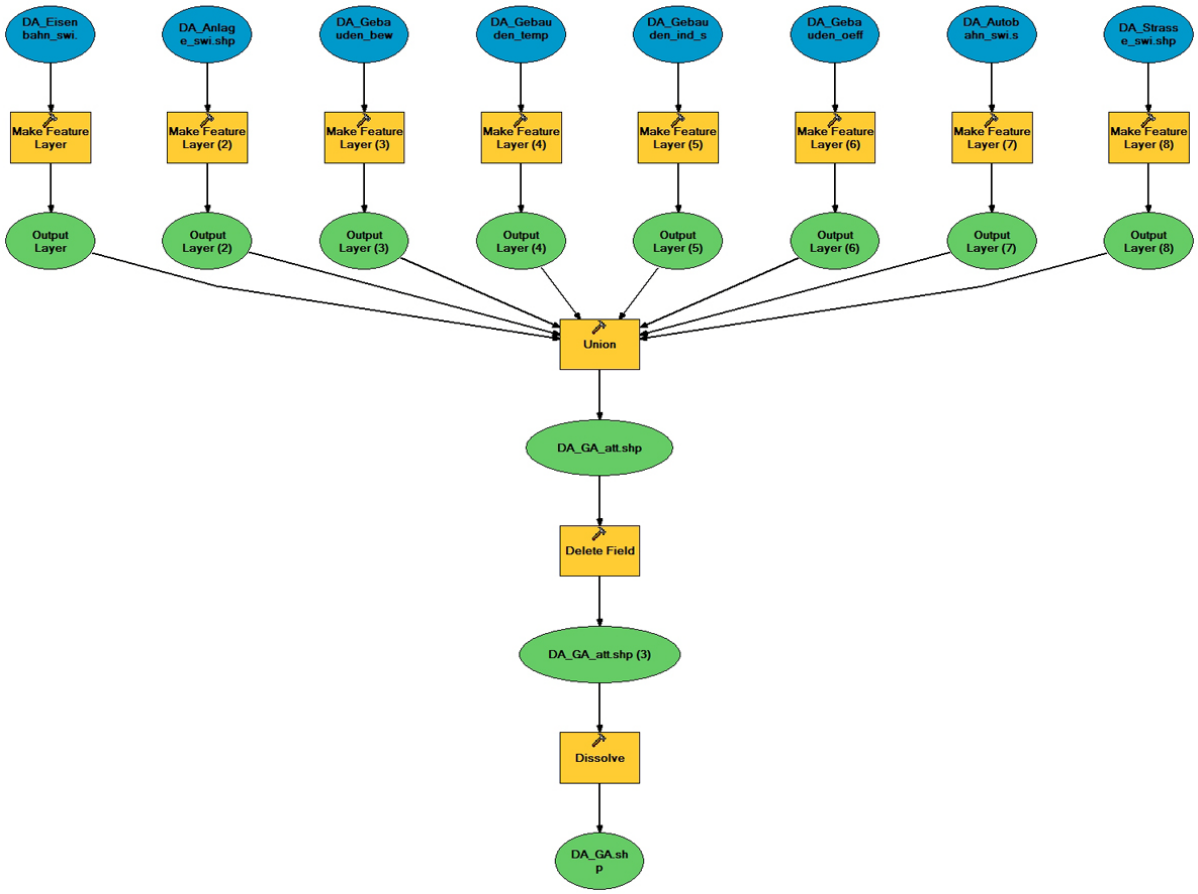
Modell: DA_GA (Toolbox: SiPro 93 DAMAGE)

Input: DA_Eisenbahn_swi, DA_Anlage_swi, DA_Gebauden_bew_swi, DA_Gebauden_temp_swi, DA_Gebauden_ind_swi, DA_Gebauden_oeff_swi, DA_Autobahn_swi und DA_Strassen_swi

Output: DA_GA_att und DA_GA.

Beschreibung des Modells:

Alle Daten zu den einzelen Schadenpotenzialkategorien wurden in einem Shapefile zusammengefügt.

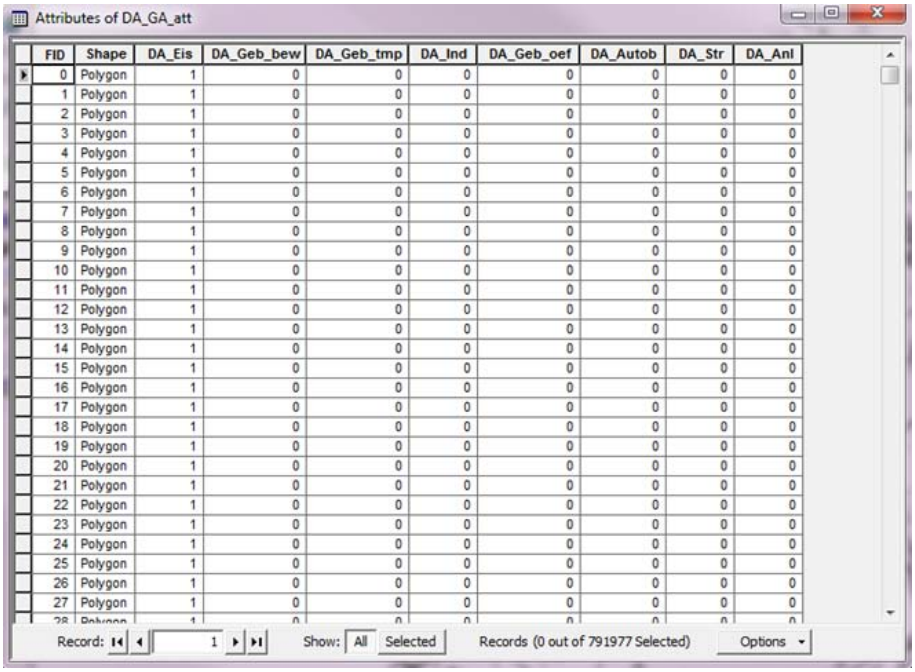


Modell 23: Schadenpotenzial für die Berechnung des Schutzwaldindex.

Beschreibung des Modells:

- **Make Feature Layer** **DA_Eisenbahn_swi**, **DA_Anlage_swi**, **DA_Gebauden_bew_swi**, **DA_Gebauden_temp_swi**, **DA_Gebauden_ind_swi**, **DA_Gebauden_oeff_swi**, **DA_Autobahn_swi** et **DA_Strassen_swi**
- **Union**: DA_Eisenbahn_swi, DA_Anlage_swi, DA_Gebauden_bew_swi, DA_Gebauden_temp_swi, DA_Gebauden_ind_swi, DA_Gebauden_oeff_swi, DA_Autobahn_swi und DA_Strassen_swi
- **Delete Field**: Drop Field: FID_DA_Eis, Id, FID_DA_Anl, Id_1, FID_DA_Geb_Id_12, FID_DA_G_1, Id_12_13, FID_DA_G_2, Id_12_1_14, FID_DA_Aut, Id_12_1_15, _FID_DA_Str, Id_12_1_16 und Id_12_1_17
- **Dissolve**: Create Multipart Feature: non checked

In der Attributtabelle des Shapefiles DA_GA_att haben alle Schadenpotenzialkategorien (Eisenbahn, Strassen, Industrie, usw.) einen Wert von 1 bei Anwesenheit und von 0 bei Abwesenheit der jeweiligen Kategorie. So kann die entsprechende Schadenpotenzialkategorie einfach gewählt werden.



FID	Shape	DA_Eis	DA_Geb_bew	DA_Geb_tmp	DA_Ind	DA_Geb_oef	DA_Autob	DA_Str	DA_Anl
0	Polygon	1	0	0	0	0	0	0	0
1	Polygon	1	0	0	0	0	0	0	0
2	Polygon	1	0	0	0	0	0	0	0
3	Polygon	1	0	0	0	0	0	0	0
4	Polygon	1	0	0	0	0	0	0	0
5	Polygon	1	0	0	0	0	0	0	0
6	Polygon	1	0	0	0	0	0	0	0
7	Polygon	1	0	0	0	0	0	0	0
8	Polygon	1	0	0	0	0	0	0	0
9	Polygon	1	0	0	0	0	0	0	0
10	Polygon	1	0	0	0	0	0	0	0
11	Polygon	1	0	0	0	0	0	0	0
12	Polygon	1	0	0	0	0	0	0	0
13	Polygon	1	0	0	0	0	0	0	0
14	Polygon	1	0	0	0	0	0	0	0
15	Polygon	1	0	0	0	0	0	0	0
16	Polygon	1	0	0	0	0	0	0	0
17	Polygon	1	0	0	0	0	0	0	0
18	Polygon	1	0	0	0	0	0	0	0
19	Polygon	1	0	0	0	0	0	0	0
20	Polygon	1	0	0	0	0	0	0	0
21	Polygon	1	0	0	0	0	0	0	0
22	Polygon	1	0	0	0	0	0	0	0
23	Polygon	1	0	0	0	0	0	0	0
24	Polygon	1	0	0	0	0	0	0	0
25	Polygon	1	0	0	0	0	0	0	0
26	Polygon	1	0	0	0	0	0	0	0
27	Polygon	1	0	0	0	0	0	0	0

Tabelle 5: Attribute der Schadenpotenzialkategorien.

.

Für eine schnellere Berechnung im Modul INTERSECT wurde das Schadenpotenzial (DA_GA) in 49 Kacheln aufgeteilt (Abbildung 16).

4.5.2 Schadenpotenzial für die Berechnung des Schadenpotenzialindex

Modell:	DA_Gew_Kt_T1 (Toolbox: SiPro 93 DAMAGE)
Input:	DA_Eisenbahn_GA_G1, DA_Eisenbahn_GA_G3, DA_Eisenbahn_GA_G5, DA_Anlage_GA_G1, DA_Anlage_GA_G3, DA_Anlage_GA_G10, DA_Gebauden_GA_G3, DA_Gebauden_GA_G5, DA_Gebauden_GA_G10, DA_Strassen_GA_G3 und DA_Strassen_GA_G5.
Output:	DA_GA_10_Diss, DA_GA_5_Diss. DA_GA_3_Diss und DA_GA_1_Diss.

Beschreibung des Modells

Das Model berechnet das Schadenpotenzial pro Gewichtung (Tabelle 3) und pro Kanton.

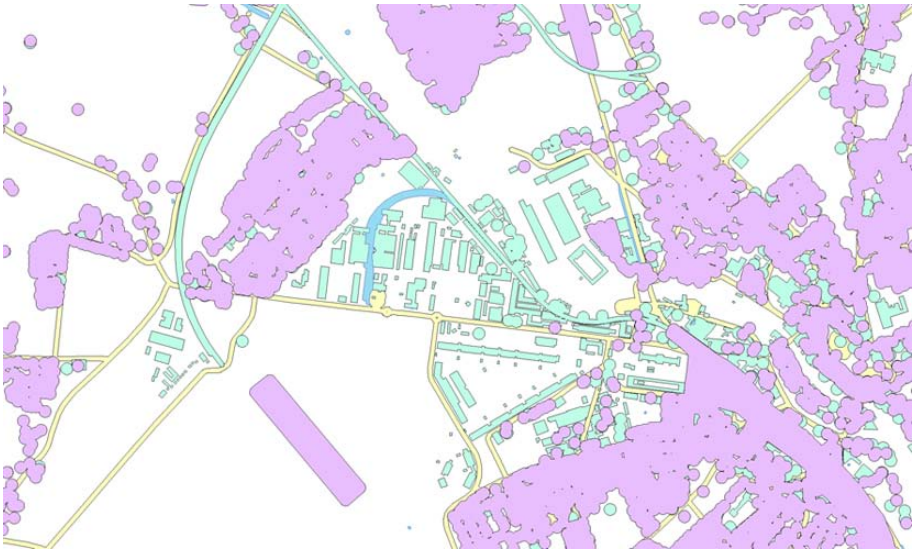
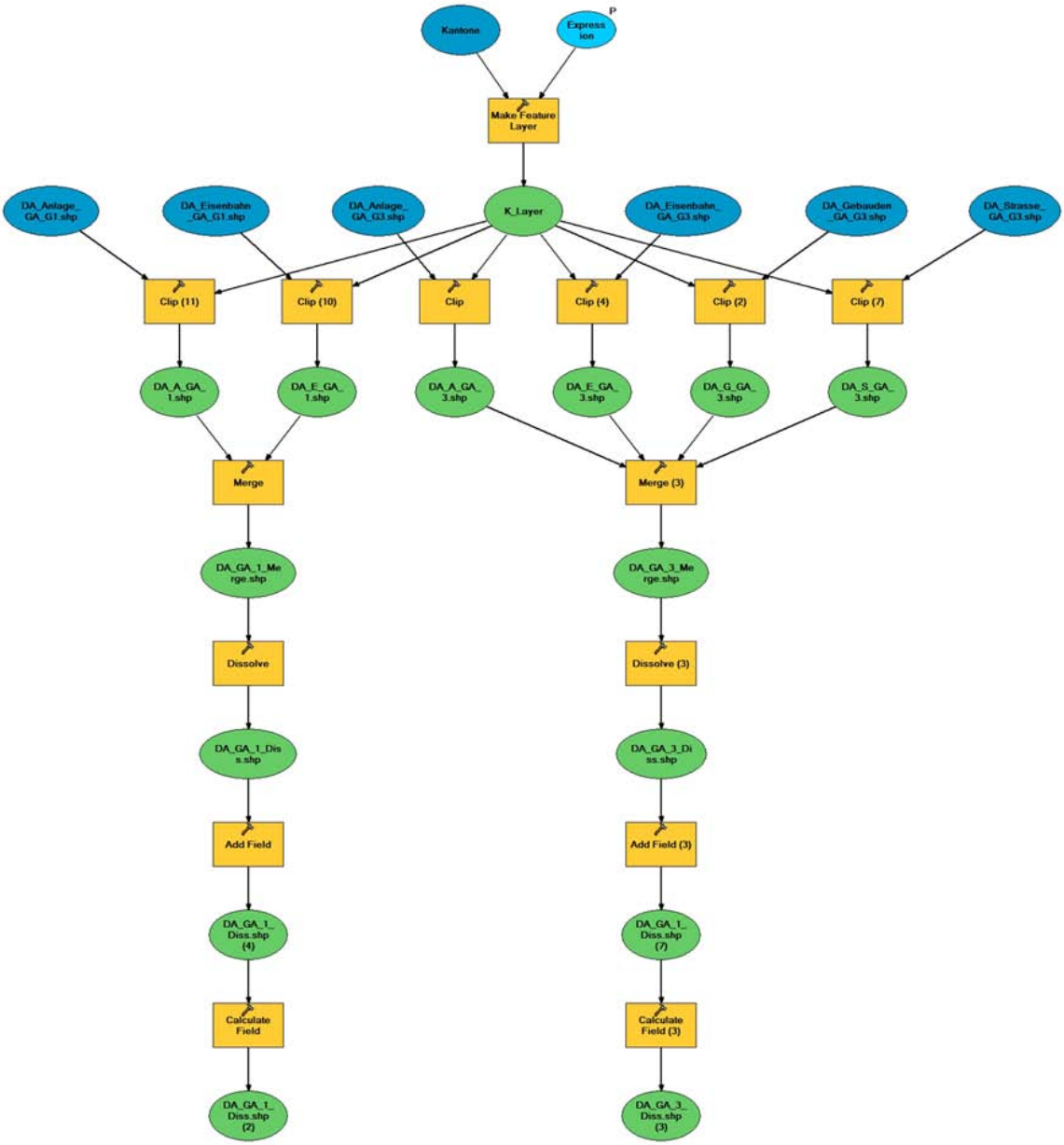


Abbildung 25: Gewichtung des Schadenpotenzials: 10 (rosa), 5 (grün), (gelb) und 1 (blau).

Der Output des Modells hat keine überlappenden Flächen. Bei überlappenden Flächen mit verschiedenen Gewichtungen wurden die Flächen mit den niedrigeren Gewichtungen automatisch gelöscht. Die obere Abbildung zeigt die verschiedenen Gewichtungen.

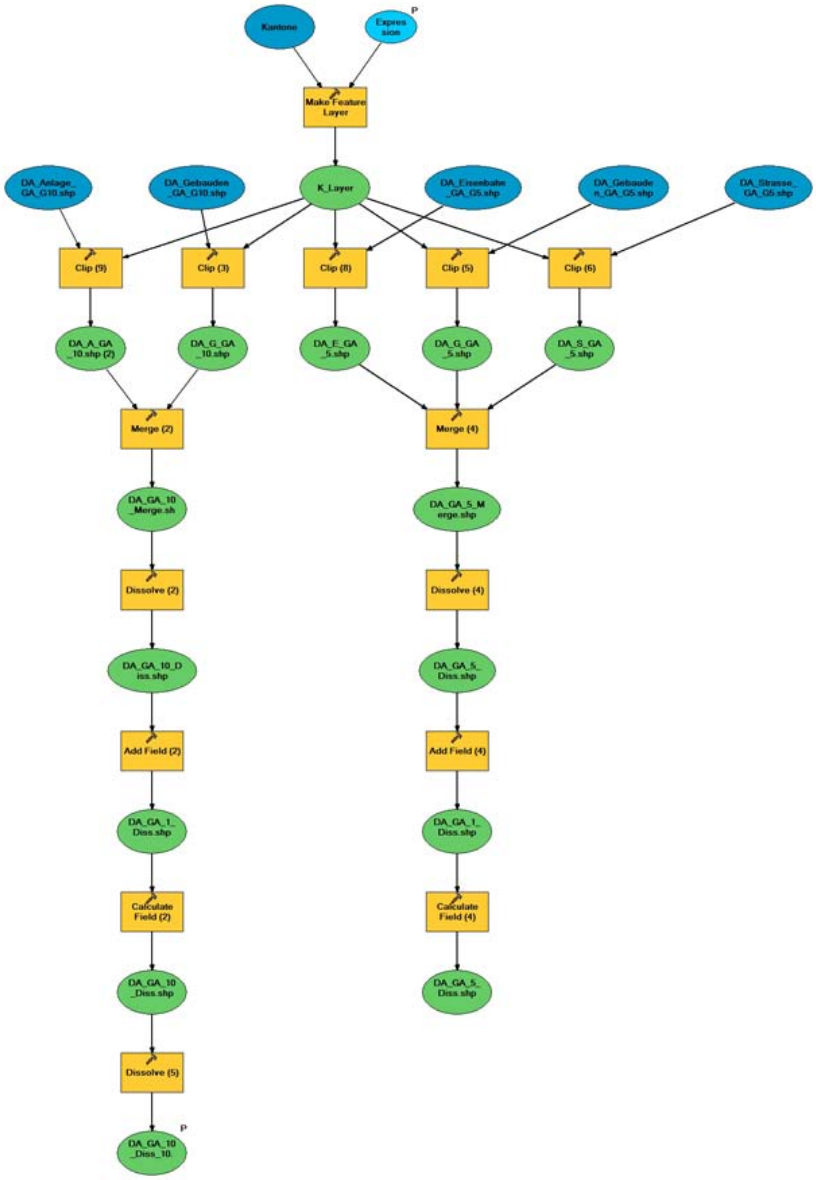


Modell 24: Schadenpotenzial für die Berechnung des Schadenpotenzialindex (Teil 1).

Beschreibung des Modells:

- **Kantone**: **Make Feature Layer**: **Expression**: (Model Parameter); “KT“ = ‘AG’
- **Clip**: Input: **DA_Anlage_GA_G3**; Clip: K_Layer
- **Clip (2)**: Input: **DA_Gebäude_GA_G3**; Clip: K_Layer
- **Clip (4)**: Input: **DA_Eisenbahn_GA_G3**; Clip: K_Layer
- **Clip (7)**: Input: **DA_Strasse_GA_G3**; Clip: K_Layer
- **Merge (3)**: DA_A_GA_3, DA_E_GA_3, DA_G_GA_3 und DA_S_GA_3
- **Dissolve (3)**: Create Multipart Feature: non checked
- **Add Field (3)**: Field Name: GEW, Field Type: SHORT
- **Calculate Field (3)**: Field Name: GEW, Expression: 3

- **Clip (10)**: Input: **DA_Eisenbahn_GA_G1**; Clip: K_Layer
- **Clip (11)**: Input: **DA_Anlage_GA_G1**; Clip: K_Layer
- **Merge**: DA_A_GA_1 und DA_E_GA_1
- **Dissolve**: Create Multipart Feature: non checked
- **Add Field**: Field Name: GEW, Field Type: SHORT
- **Calculate Field**: Field Name: GEW, Expression: 1

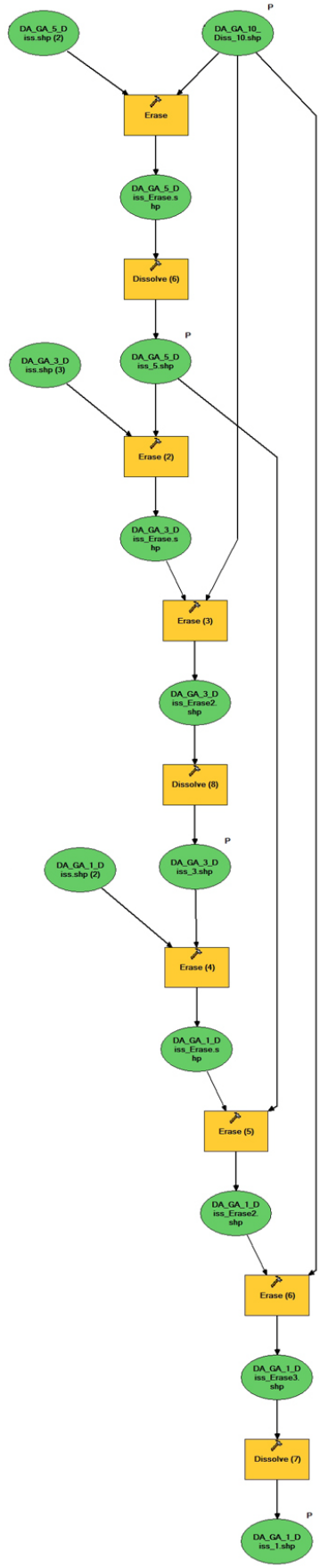


Modell 25: Schadenpotenzial für die Berechnung des Schadenpotenzialindex (Teil 2).

Beschreibung des Modells:

- **Kantone**: Make Feature Layer: **Expression**: (Model Parameter); "KT" = 'AG'
- **Clip (3)**: Input: **DA_Gebäude_GA_G10**; Clip: K_Layer
- **Clip (9)**: Input: **DA_Anlage_GA_G10**; Clip: K_Layer
- **Merge (2)**: DA_A_GA_10 und DA_G_GA_10
- **Dissolve (2)**: Create Multipart Feature: non checked
- **Add Field (2)**: Field Name: GEW, Field Type: SHORT
- **Calculate Field (2)**: Field Name: GEW, Expression: 10
- **Dissolve (5)**: Input: DA_GA_10_Diss; Dissolve Field(s): GEW: checked; Create Multipart Feature: checked

- **Clip (5)**: Input: **DA_Gebäude_GA_G5**; Clip: K_Layer
- **Clip (6)**: Input: **DA_Strasse_GA_G5**; Clip: K_Layer
- **Clip (8)**: Input: **DA_Eisenbahn_GA_G5**; Clip: K_Layer
- **Merge (4)**: DA_E_GA_5, DA_G_GA_5 und DA_S_GA_5
- **Dissolve (4)**: Create Multipart Feature: non checked
- **Add Field (4)**: Field Name: GEW, Field Type: SHORT
- **Calculate Field (4)**: Field Name: GEW, Expression: 5



Modell 26: Schadenpotenzial für die Berechnung des Schadenpotenzialindex (Teil 3).

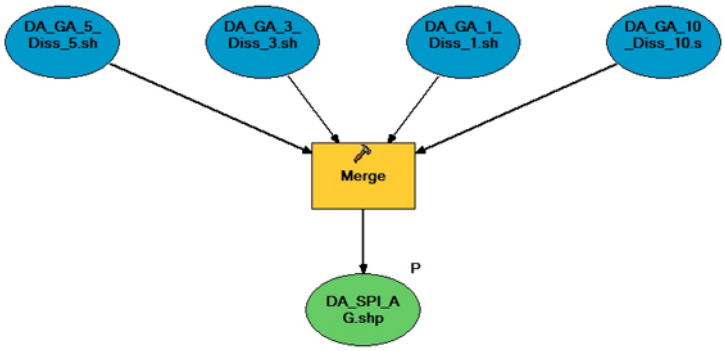
Beschreibung des Modells:

- **Erase**: Input: DA_GA_5_Diss, Erase: DA_GA_10_Diss
- **Dissolve (6)**: Input: DA_GA_5_Erase; Dissolve Field(s): GEW: checked; Create Multipart Feature: checked
- **Erase (2)**: Input: DA_GA_3_Diss, Erase: DA_GA_5_Diss
- **Erase (3)**: Input: DA_GA_3_Erase, Erase: DA_GA_10_Diss
- **Dissolve (8)**: Input: DA_GA_3_Erase_Erase; Dissolve Field(s): GEW: checked; Create Multipart Feature: checked
- **Erase (4)**: Input: DA_GA_1_Diss, Erase: DA_GA_3_Diss
- **Erase (5)**: Input: DA_GA_1_Erase, Erase: DA_GA_5_Diss
- **Erase (6)**: Input: DA_GA_1_Erase_Erase, Erase: DA_GA_10_Diss
- **Dissolve (7)**: Input: DA_GA_1_Erase_Erase_Erase; Dissolve Field(s): GEW: checked; Create Multipart Feature: checked

Modell: DA_Gew_Kt_T2 (Toolbox: SiPro 93 DAMAGE)

Input: DA_GA_10_Diss, DA_GA_5_Diss. DA_GA_3_Diss und DA_GA_1_Diss.

Output: DA_SPI_AG.



Modell 27: Schadenpotenzial für die Berechnung des Schadenpotenzialindex (Teil 4).

Beschreibung des Modells:

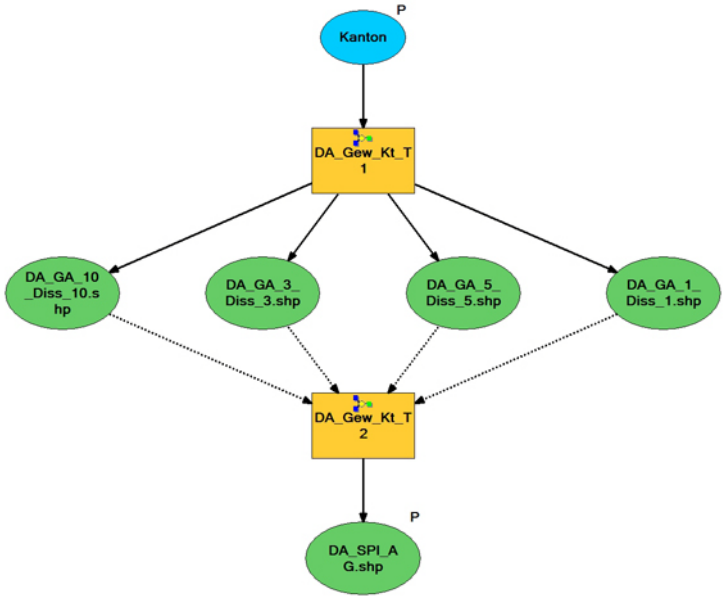
- **Merge**: DA_GA_10_diss, DA_GA_5_diss, DA_GA_3_diss und DA_GA_1_diss

Die zwei obere Schritte (DA_Gew_Kt_T1 und DA_Gew_Kt_T2) können in einem einzelnen Modell (DA_Gew_Kt) implementiert werden

Modell: DA_Gew_Kt (Toolbox: SiPro 93 DAMAGE)

Input: DA_Gew_Kt_T1 und DA_Gew_Kt_T2.

Output: DA_SPI_AG bis DA_SPI_ZH.



Modell 28: Berechnung des Schadenpotenzialindex pro Kanton (Zusammenzug beider Modelle)

5 Relevante Prozesse (INTERSECT)

5.1 Lawine: relevante Anrissgebiete

5.1.1 Beschreibung der Modellierung

Die Lawinen sind in drei Kategorien geteilt: grosse, mittlere und kleine (Kapitel 3.1). Bei den Lawinen der Klassen „klein“ und „mittel“ wurden die Anrissgebiete auf folgende Weise bestimmt: Wenn ein Lawinenperimeter auf ein Schadenpotenzial traf, wurde das entsprechende Anrissgebiet als relevant bezeichnet (Abbildung 27).

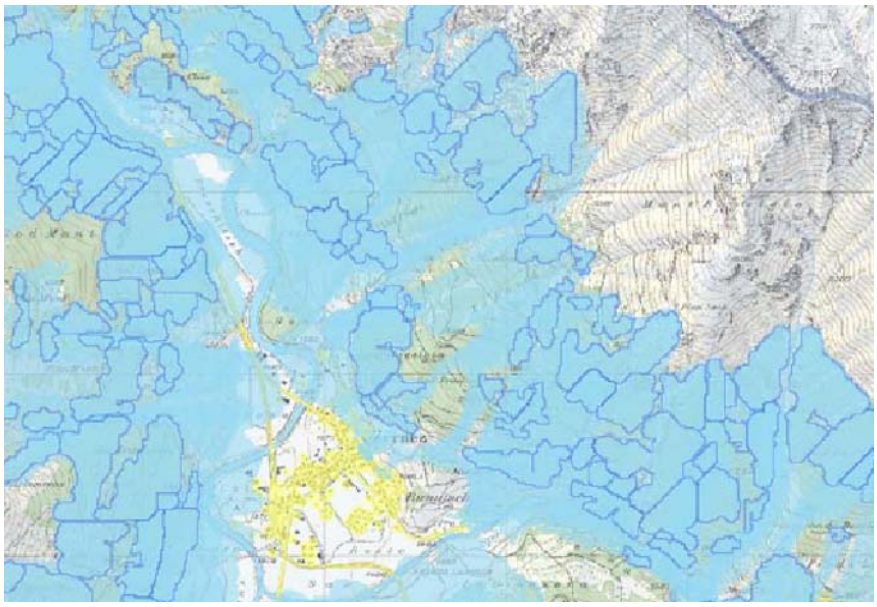


Abbildung 26: Anrissgebiete (dunkelblau) der Lawinen, die innerhalb der heutigen Waldfläche anreissen, und dazu gehörenden Lawinenperimeter (hellblau). In Gelb ist das Schadenpotenzial dargestellt.

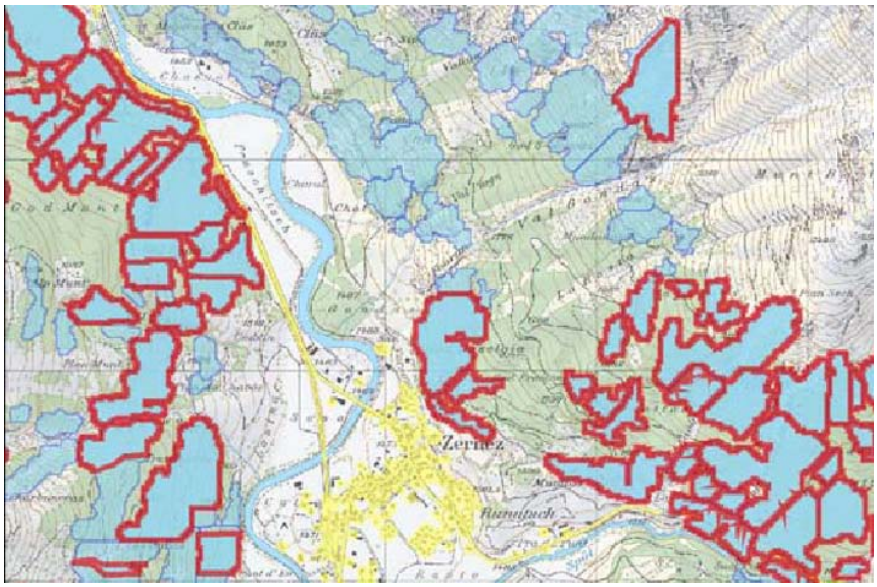


Abbildung 27: Relevante Anrissgebiete (rot), die auf ein Schadenpotenzial treffen.

Da sich Anrissgebiete der Klasse gross unter Umständen über ganze Talflanken erstrecken können und das Schadenpotenzial nur an einem Ort konzentriert ist, wurde eine Methode entwickelt, welche grosse Anrissgebiete weiter unterteilt. So können die relevanten Flächen der Anrissgebiete definiert werden, die zur effektiven Gefährdung des Schadenpotenzials beitragen (Teil der Lawine: Abbildung 28 und Schadenpotenzial am Gegenhang: Abbildung 29)

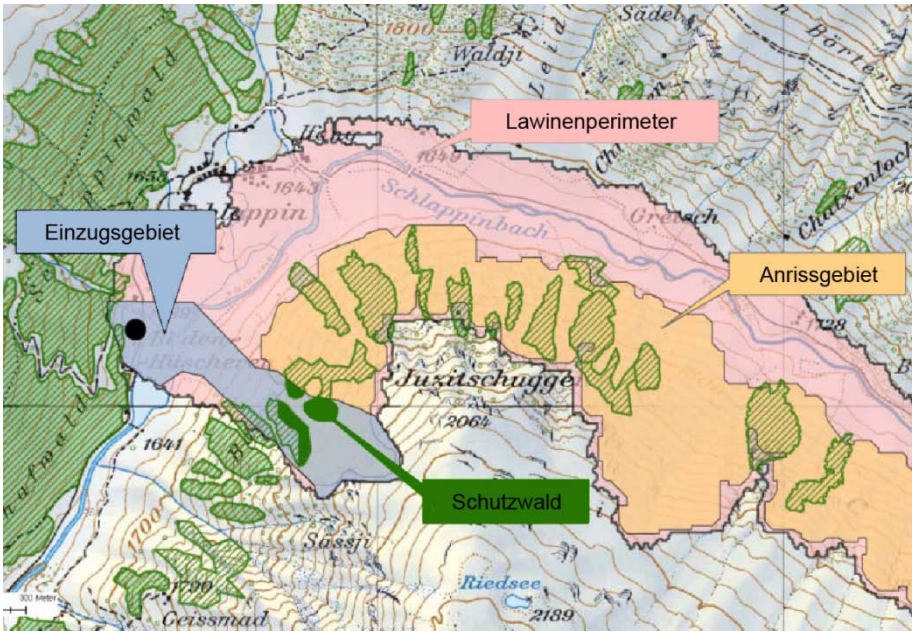


Abbildung 28: Ausgehend aus einem punktuellen Schadenpotenzial (schwarzer Punkt), wird mittels DHM10 der Teil des Lawinenperimeter ermittelt, der schadenrelevant ist (Einzugsgebiet in blau) (Klosters GR).

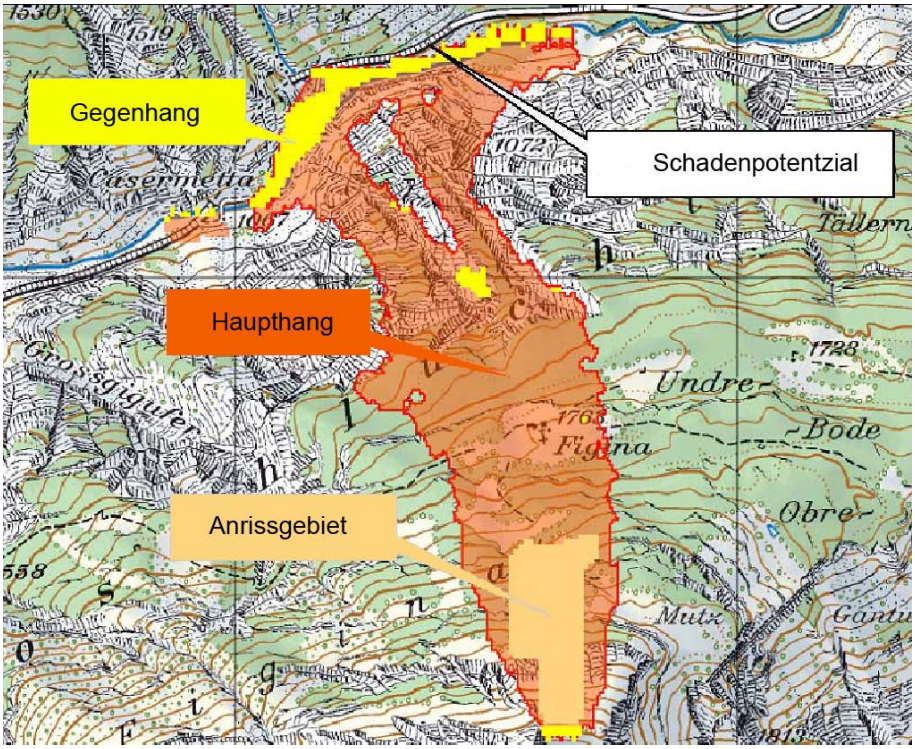


Abbildung 29: Ermittlung der schadenrelevanten Anrissgebiete „gross“ (Zwischbergen VS).

5.1.2 Schadenpotenzial für die Lawinengebiete

Modell:	DA_Lawine_pro_Gebiet (Toolbox: SiPro 93 IN Lawine)
Input:	BA_Lawine_Gebiete und DA_GA
Output:	30 Shapes: DA_adu, DA_bad … bis DA_wil.

Beschreibung des Modells

Das Schadenpotenzial wurde mit einem Puffer von 10 km versehen und auf die Lawinengebiete (Abbildung 17) aufgeteilt. Somit konnte sichergestellt werden, dass das Schadenpotenzial gut abgedeckt ist.

Das Modell wurde mit einer List of values erstellt, wie aus Abbildung 30 ersichtlich ist.

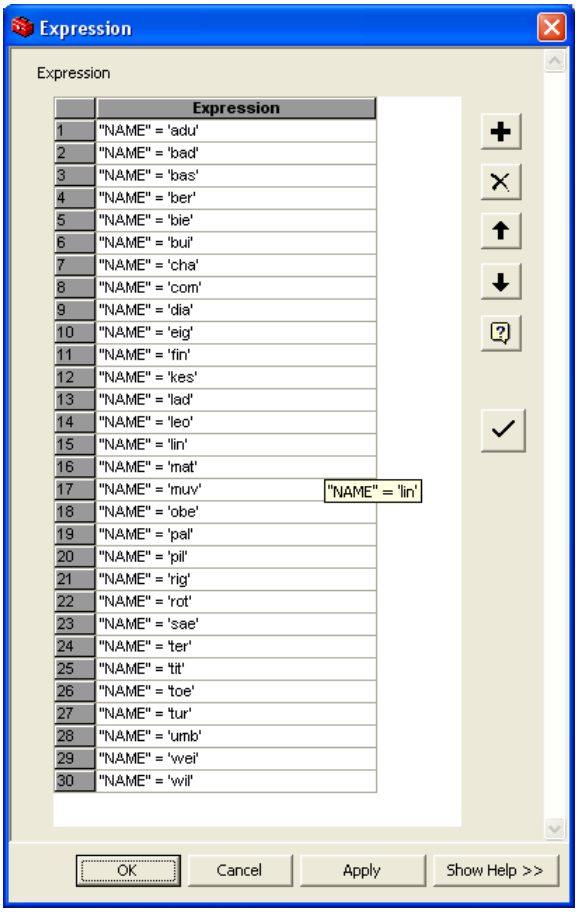
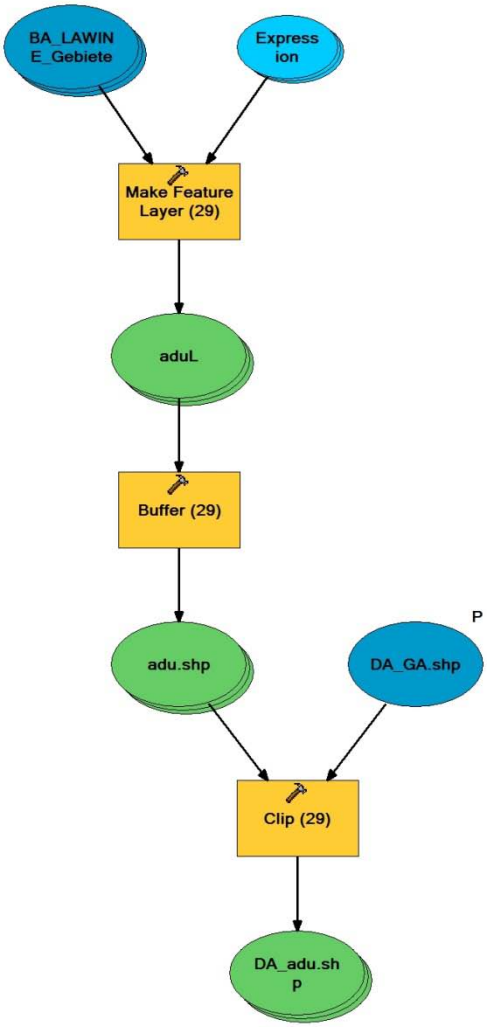


Abbildung 30: List of value für das Modell 29.



Modell 29: Schadenpotenzial pro Lawinengebiet.

Beschreibung des Modells:

- **BA_LAWINE_Gebiete**: **Make Feature Layer**: Expression: voir **Erreur ! Source du renvoi introuvable.**
- **Buffer**: Distance: 10 km, End Type: ROUND
- **DA_GA**: **Make Feature Layer**
- **CLIP**: Input Features: DA_GA, Clip Features: Lawinengebiet

5.1.3 Relevante Anrissgebiete

Der Prozess Lawine ist im Kapitel 3.1 beschrieben und die Vektordaten sind unter D:\SilvaProtect\BASIC\Input gespeichert. Um die Anrissgebiete zu berechnen, wurden verschiedene Skripte in Python (Version 2.5) programmiert und verwendet (Kapitel 9.3).

- pp_Launch
- pp_proc
- pp_Programm
- pp_Settings
- pp_util_GIS

Die Skripte starten automatisch nach dem Start von pp_Launch; nur das Skript pp_Settings muss noch wie folgt angepasst werden (Abbildung 31) .

- src_Workspace: Ort wo die benötigten Daten gespeichert sind
- tar_Folder: Ort wo die generierten Daten gespeichert werden
- gebiet: Bearbeitetes Lawinengebiet
- DamLayer: Daten für das Schadenpotenzial aus dem Modell DA_Lawine_pro_Gebiet
- dtm: Benutztes Geländemodell

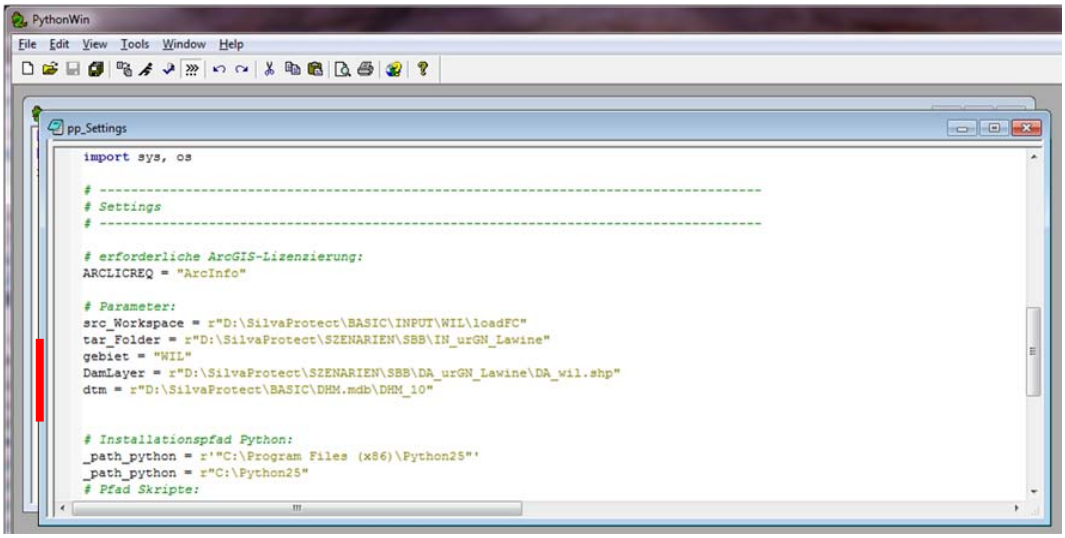


Abbildung 31: Script pp_Settings mit den nötigen Parametern.

Zusätzliche GIS-Arbeiten:

Die Ergebnisse sind in einer neuen Geodatabase gespeichert und sehen so aus: wiln_rel_medium, wiln_rel_small et wiln_sw_large.

- **Merge**: Input: adun_rel_small, adun_rel_medium, bis wiln_rel_medium und wiln_rel_large
- **Dissolve:** Create Multipart Feature: non checked

In der Attributtabelle wurde eine neue Kolonne mit dem Name „LAW“ und dem Wert „1“ eingefügt.

- **Add Field**: Field Name: LAW, Field Type: SHORT
- **Calculate Field**: Field Name: LAW, Expression: 1
- **Delete Field**: alle unnötigen Felder werden gelöscht.

Das Schlussresultat ist das Shape IN_LAWINE_CH.

5.2 Relevante Sturztrajektorien

5.2.1 Beschreibung der Modellierung

Für diesen Prozess wurden alle Sturztrajektorien, die auf ein Schadenpotenzial treffen, selektioniert (Abbildung 32)auf 10 Meter gepuffert (Abbildung 33) und mit einer Umhüllenden „aufgelöst“ (Abbildung 34).

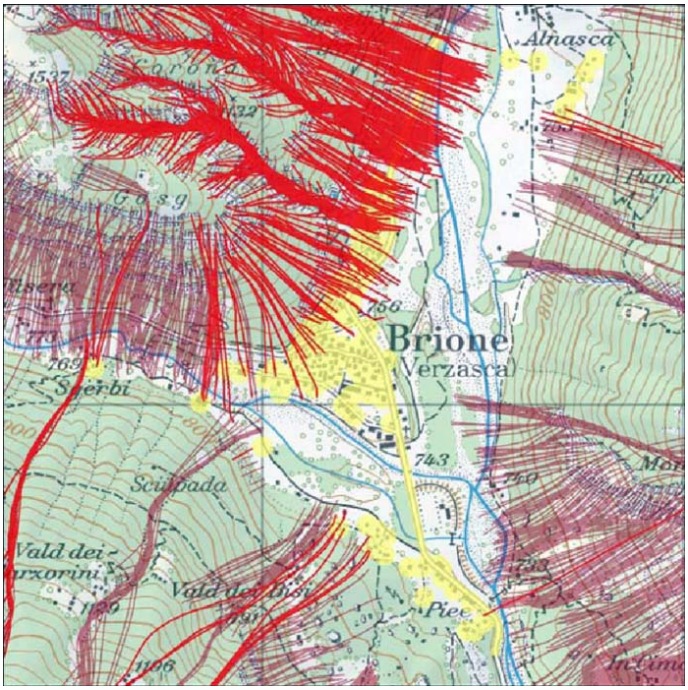


Abbildung 32: Ermittlung der schadenrelevanten Sturztrajektorien (hellrot).

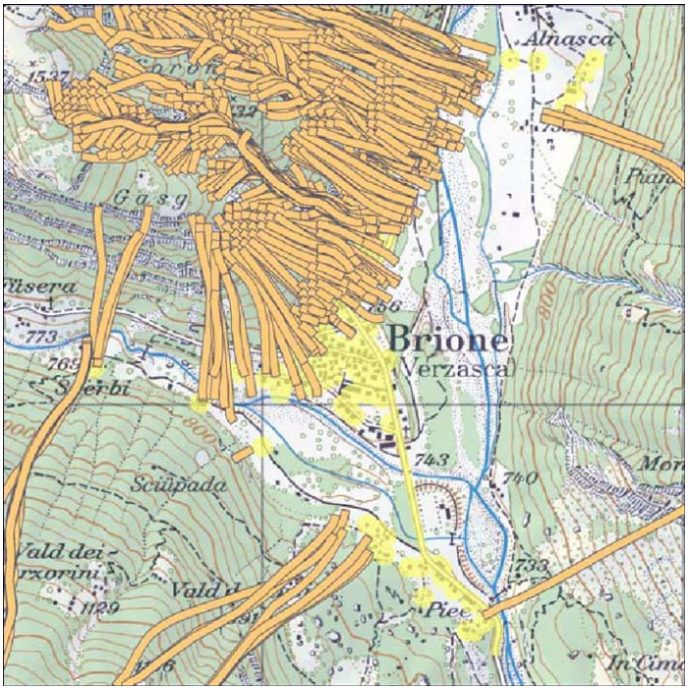


Abbildung 33: Pufferung der schadenrelevanten Sturztrajektorien (orange).

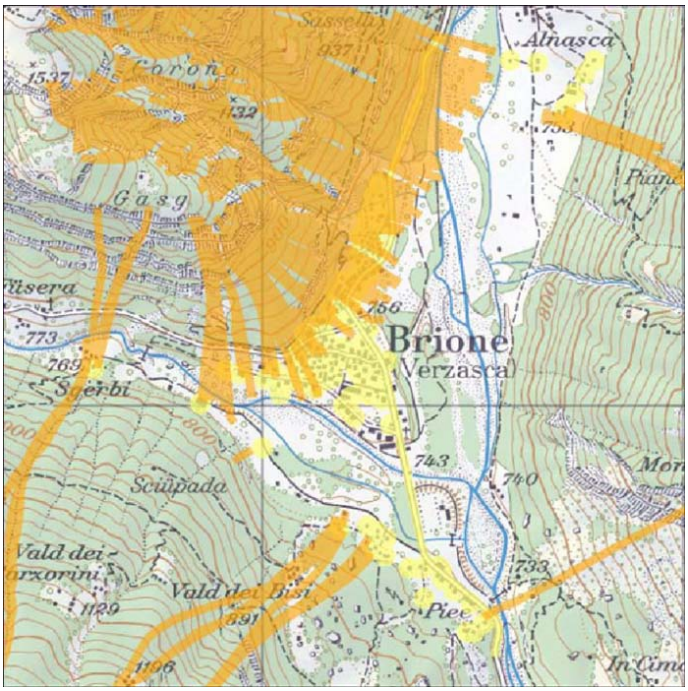


Abbildung 34: Bildung einer Umhüllenden der schadenrelevanten Sturztrajektorien (orange).

5.2.2 Relevante Sturztrajektorien für eine Kachel

Modell: IN Sturz07 (Toolbox: SiPro 93 IN STURZ)

Input: EV_Sturz_07, TILES10 und DA_GA_07

Output: 4 Shapes: 7bd, 7da, 7db und 7dd.

Beschreibung des Modells

Dieses Modell berechnet alle relevanten Trajektorien der Kachel 7 gemäss Abbildung 16. Das Modell kann für alle Schadenpotenzialkategorien benutzt werden: Eisenbahnnetz, Anlagen, Gebäude, etc. Es wurde erstellt mit Hilfe von "List of value" und die folgenden Abbildungen zeigen die dabei benützten Kriterien.

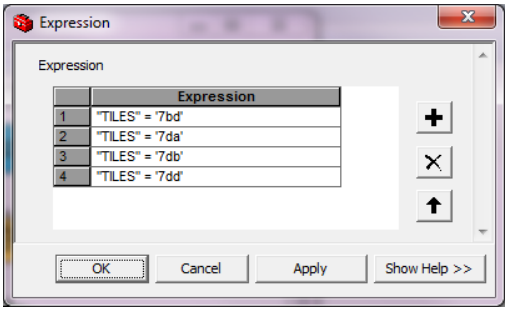


Abbildung 35: Expression für das Modell 30.

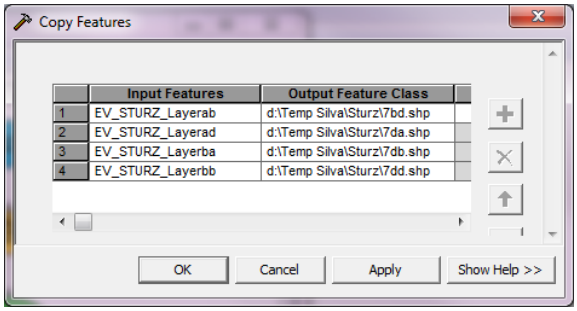
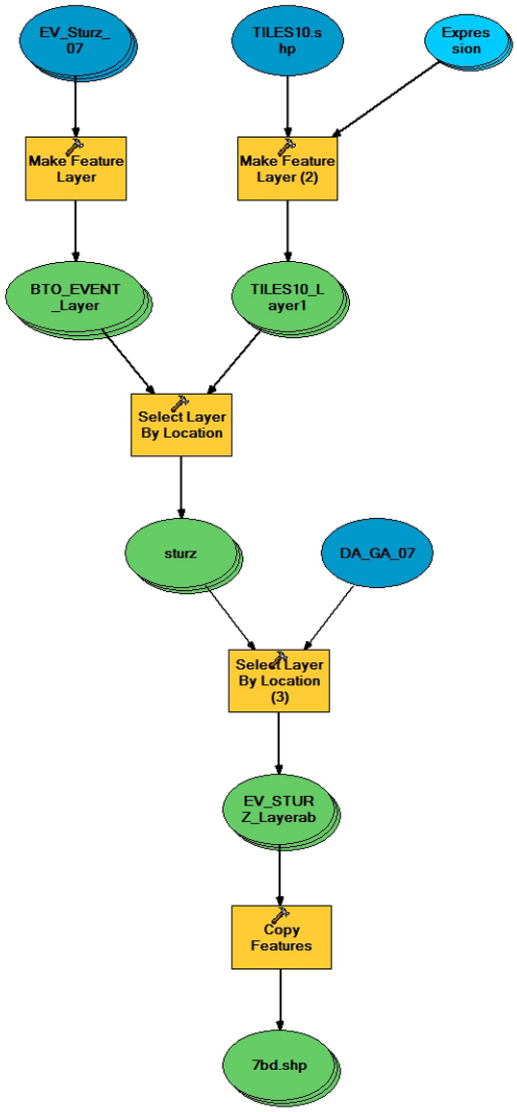


Abbildung 36: Resultate mit dem Modell 30.



Modell 30: Sturztrajektorien, die auf ein Schadenpotenzial treffen.

Beschreibung des Modells:

- **EV_STURZ_07**: **Make Feature Layer**, Properties: List of values
- **TILES10**: **Make Feature Layer**, Expression: (Abbildung 35)
- **Select Layer By Location**: Input: EV_STURZ, Selecting: TILES10
- **Select Layer By Location**: Input: Sturz, Selecting: **DA_GA_07**, Selection type: SUB-SET_SELECTION
- **Copy Features**: (Abbildung 36)

5.2.3 Berechnung der relevanten Trajektorien für die ganze Schweiz

Modell:	IN STURZ (Toolbox: SiPro 93 IN STURZ)
Input:	Modelle IN Sturz01, IN Sturz02, … bis IN Sturz49 EV_STURZ_01 bis EV_STURZ_49, TILES10 und DA_GA_01 bis DA_GA_49
Output:	ungefähr 500 Shapes: 1bc, 1bd, bis 49cc und 49cd.

Beschreibung des Modells

Für alle Kachel (Abbildung 16) wurde ein einziges Modell wie oben beschreiben erstellt. Die 49 Modelle sind nachher in einem Gesamtmodell eingefügt. Damit kann die Berechnung selbständig laufen.

Modell 31: Relevante Sturztrajektorien für die ganze Schweiz.

Zusätzliche GIS-Arbeiten:

Für alle Kacheln wurden die relevanten Sturztrajektorien gepuffert und aufgelöst mit dem Modell 1: Umwandlung von Linien in Polygone. Danach wurden alle bearbeiteten Kacheln zusammengesetzt.

- **Dissolve: DR01** bis **DR49**
- **Merge**: Input: 1bc_diss, 1bd_diss, bis 49cc_diss und 49cd_diss
- **Dissolve:** Create Multipart Feature: non checked

In der Attributtabelle wurde eine neue Kolonne mit dem Name „STURZ“ und dem Wert „1“ eingefügt .

- **Add Field**: Field Name: STURZ, Field Type: SHORT
- **Calculate Field**: Field Name: STURZ, Expression: 1
- **Delete Field**: alle unnötigen Felder werden gelöscht.

Wenn die Berechnung für alle Kacheln fertig ist, werden alle Shapefiles zusammengefügt (MERGE) und aufgelöst (DISSOLVE). **Das Schlussresultat ist das Shape IN_STURZ_CH**.

5.3 Relevante Hangmurentrajektorien

5.3.1 Beschreibung der Modellierung

Für diesen Prozess werden alle Hangmurentrajektorien, die auf ein Schadenpotenzial treffen, selektioniert (Abbildung 37), auf 10 Meter gepuffert (Abbildung 38) und mit einer Umhüllenden „aufgelöst“ (Abbildung 39).

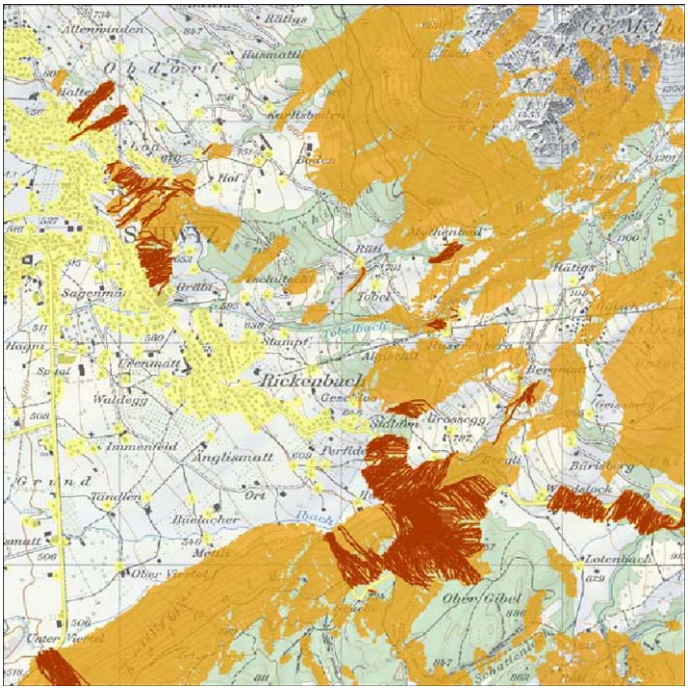


Abbildung 37: Ermittlung der schadenrelevanten Hangmurentrajektorien (braun).

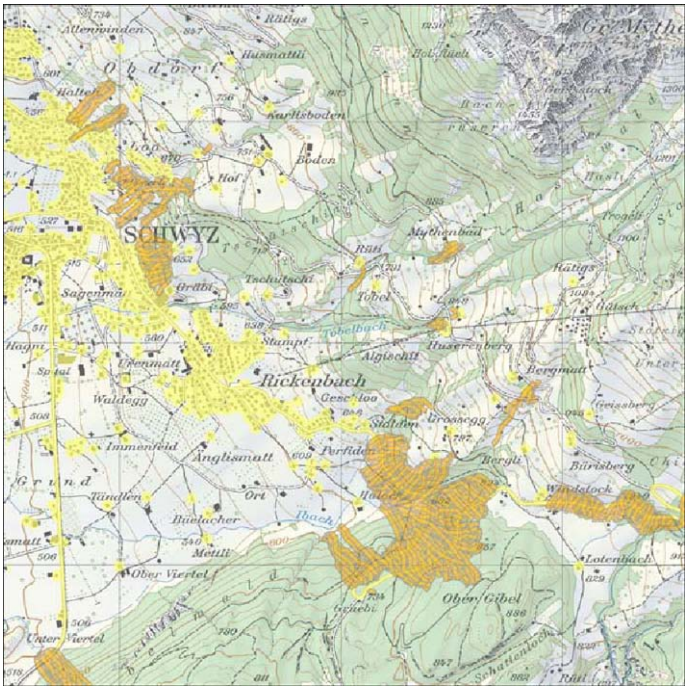


Abbildung 38: Pufferung der schadenrelevanten Hangmurentrajektorien (orange).

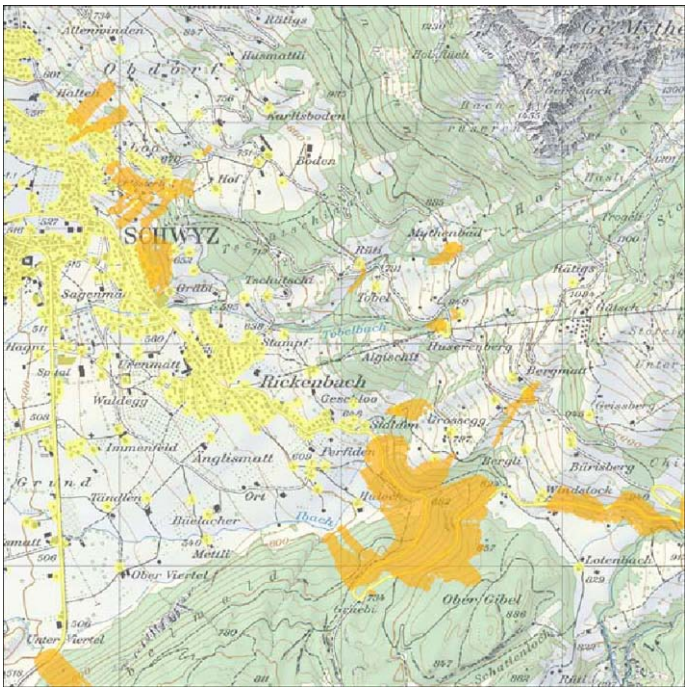


Abbildung 39: Bildung einer Umhüllenden der schadenrelevanten Hangmurentrajektorien (orange).

5.3.2 Relevante Hangmurentrajektorien für eine Kachel

Modell:	IN HM01 (Toolbox: SiPro 93 IN HM)
Input:	EV_HANGMURE_01, TILES10 et DA_GA_01
Output:	30 Shapes: 1bc, 1bd, 1cc, 1cd, 1da, 1db, 1dc et 1dd.

Beschreibung des Modells

Dieses Modell berechnet alle relevanten Trajektorien der Kachel 1 gemäss Abbildung 16. Das Modell kann für alle Schadenpotenzialkategorien benutzt werden: Eisenbahnnetz, Anlagen, Gebäude, etc. Es wurde erstellt mit Hilfe von "List of value" und die folgenden Abbildungen zeigen die dabei benützten Kriterien.

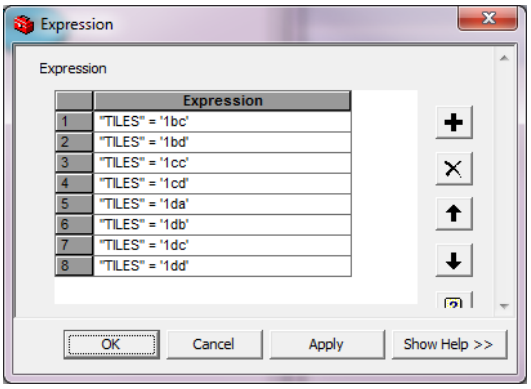


Abbildung 40: Expression für das Modell 32.

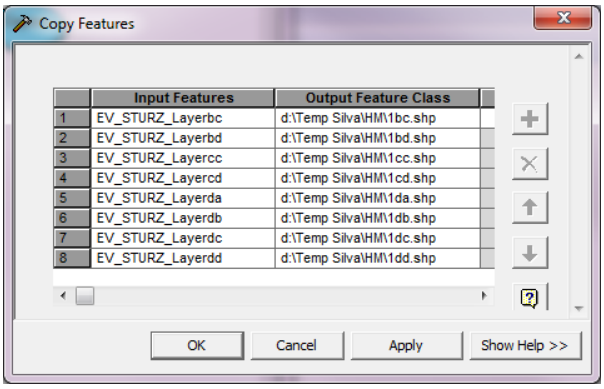
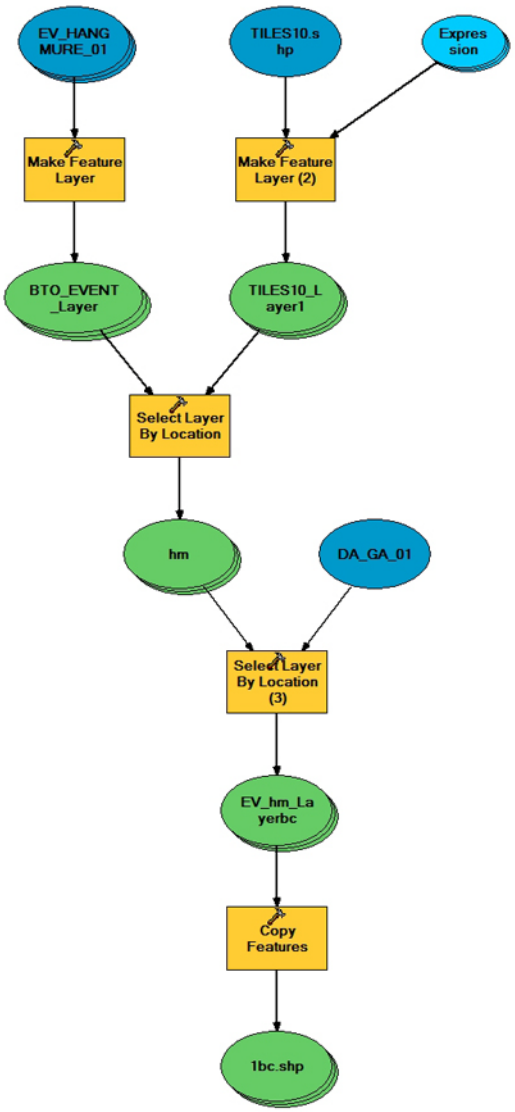


Abbildung 41: Resultate mit dem Modell 32.



Modell 32: Relevante Hangmure.

Beschreibung des Modells:

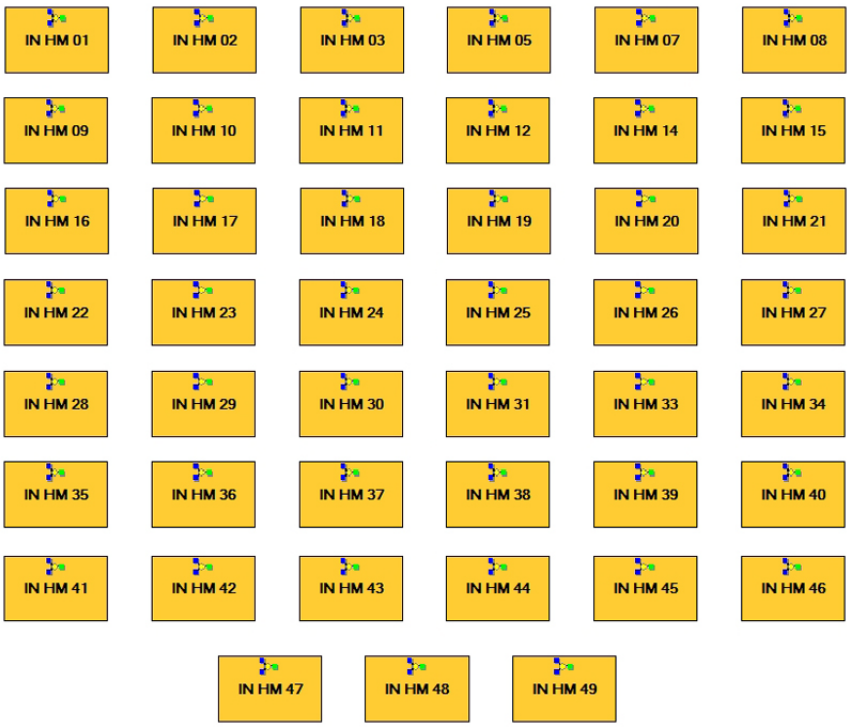
- **EV_HANGMURE_01**: **Make Feature Layer**, Properties: List of values
- **TILES10**: **Make Feature Layer**, Expression: (Abbildung 40)
- **Select Layer By Location**: Input: EV_HANGMURE, Selecting: TILES10
- **Select Layer By Location**: Input: Sturz, Selecting: **DA_GA_01**, Selection type: SUB-SET_SELECTION
- **Copy Features**: (Abbildung 41)

5.3.3 Berechnung aller Kacheln zusammen

Modell:	IN HM (Toolbox: SiPro 93 IN HM)
Input:	Modelle IN HM 01, IN HM 02, … bis IN HM 49 EV_HANGMURE_01 bis EV_ HANGMURE _49, TILES10 und DA_GA_01 bis DA_GA_49
Output:	ungefähr 500 Shapes: 1bc, 1bd, bis 49cc und 49cd.

Beschreibung des Modells

Für alle Kachel (Abbildung 16) wurde analog zu Modell 32 ein einziges Modell erstellt. Die 49 einzelnen Modelle werden nachher in einem Gesamtmodell eingefügt. Damit kann die Berechnung selbständig laufen.



Modell 33: Relevante Hangmuren für die ganze Schweiz.

Zusätzliche GIS-Arbeiten:

Für alle Kachen wurden die relevante Sturztrajektorien gepuffert und aufgelöst mit dem Modell 1: Umwandlung von Linien in Polygone. Danach wurden alle bearbeiteten Kacheln zusammengesetzt.

- **Dissolve: DR01** bis **DR49**
- **Merge**: Input: 1bc_diss, 1bd_diss, bis 49cc_diss und 49cd_diss
- **Dissolve:** Create Multipart Feature: non checked

In der Attributtabelle wurde eine neue Kolonne mit dem Name „HM“ und dem Wert „1“ eingefügt.

- **Add Field**: Field Name: HM, Field Type: SHORT
- **Calculate Field**: Field Name: HM, Expression: 1
- **Delete Field**: alle unnötigen Felder werden gelöscht.

Wenn die Berechnung für alle Kacheln fertig ist, werden alle Shapefiless zusammengefügt(MERGE) und aufgelöst (DISSOLVE). **Das Schlussresultat ist das Shape IN_HM_CH**.

5.4 Relevante Gerinneprozesse

Wie aus Abbildung 42 ersichtlich ist, werden die Gerinneprozesse ausgehend vom relevanten Gerinnenetz modelliert. Das relevante Gerinnenetz beschreibt die Gerinne, bei welchen ein Gefahrenprozess auf ein relevantes Schadenpotenzial gemäss Kapitel 4 treffen kann.

Dieses Gerinnenetz wird primär bestimmt anhand der Prozesse Übersarung und Murgang sowie des Schadenpotenzials.

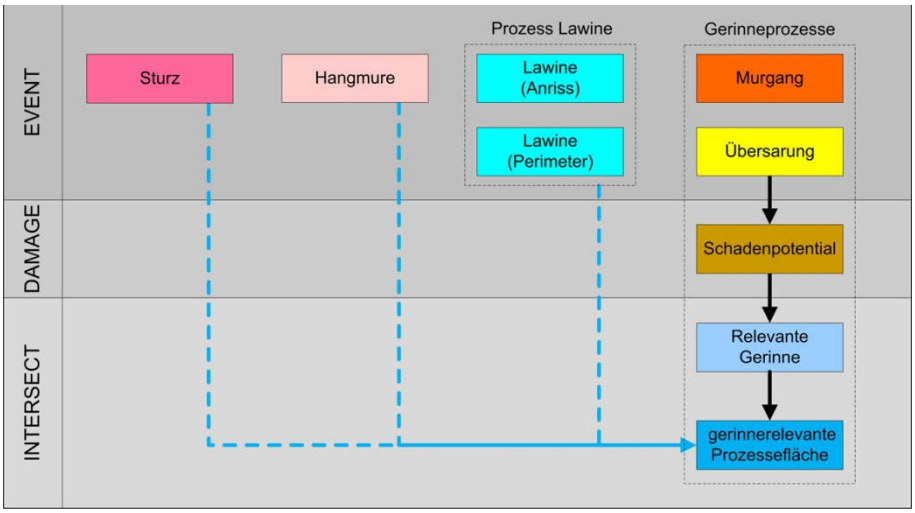


Abbildung 42: Beschreibung für die Modellierung der relevanten Gerinnen.

Kapitel 5.4 teilt sich auf in 2 Teile: Im ersten Teil wird beschrieben, wie das relevante Gerinnenetz mit Hilfe der Gefahrenprozesse Übersarung (5.4.1) und Murgang (5.4.2) festgelegt wird. Im zweiten Teil folgt ein Beschrieb der für dieses Gerinnenetz relevanten Naturgefahrenprozesse (5.4.3).

5.4.1 Bestimmung der relevanten Gerinne mit Hilfe des Prozesses Übersarung

5.4.1.1 Beschreibung der Modellierung

Bei Übersarungen wird Geschiebe transportiert, wobei sich die Feststoffe (Erde, Stein, Geröll, Schutt) oft flächendeckend ausserhalb des Gerinnes ablagern. Übersarungsprozesse, welche auf ein Schadenpotenzial treffen, werden selektioniert (Abbildung 43) und ihr Startpunkt wird bestimmt (Abbildung 44). Mit Hilfe der Ausbruchspunkte können mit einer eigens entwickelten semi-automatischen GIS-Methode die relevanten Gerinne bestimmt werden (Abbildung 45).

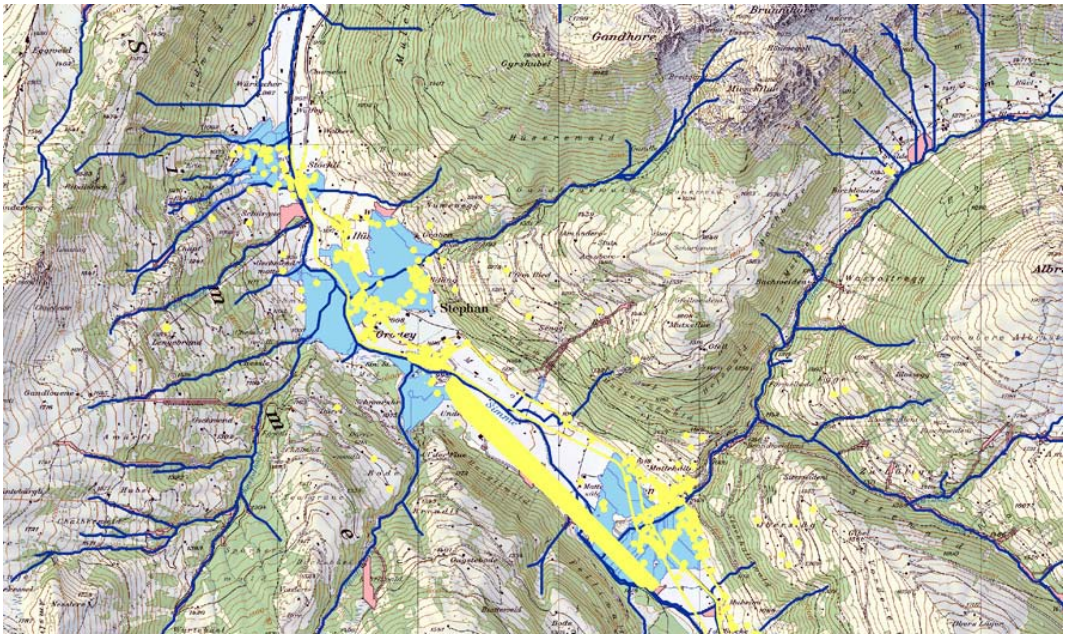


Abbildung 43: Relevante (blau) und nicht-relevante (rosa) Übersarungsflächen und das Schadenpotenzial (gelb)

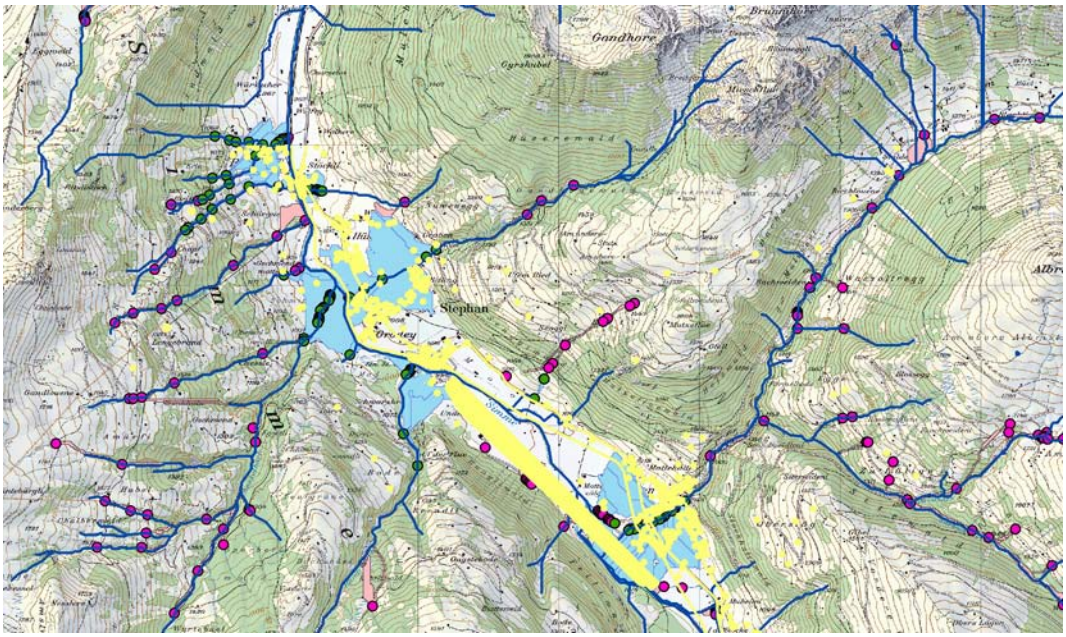


Abbildung 44: Relevante (grün) und nicht-relevante (violett) Startpunkte für die Übersarungen und Schadenpotenzial (gelb).

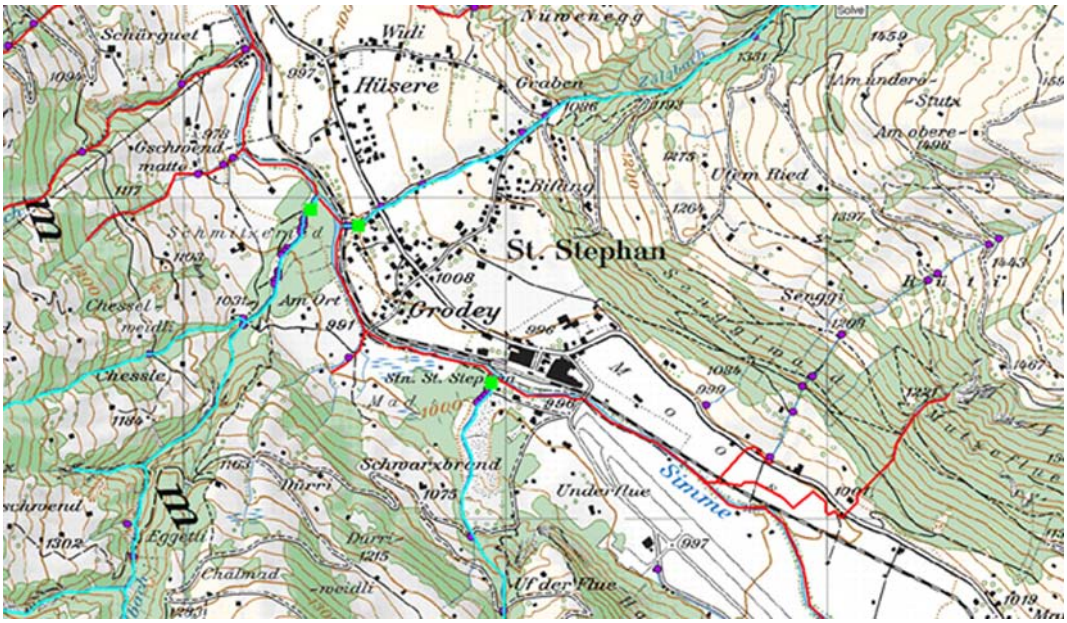
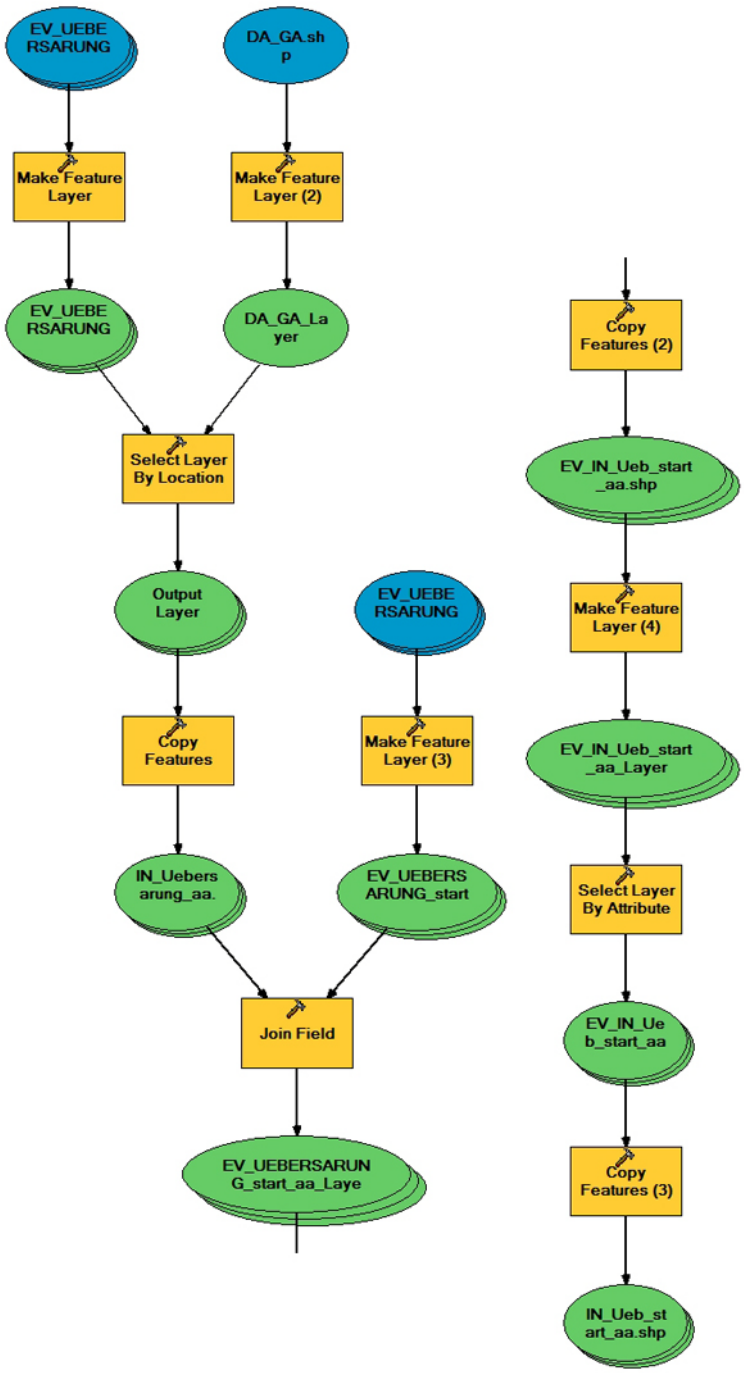


Abbildung 45: Bestimmung der relevanten Gerinnen mit dem GIS-Tool "Utility Network Analyst".

5.4.1.2 Relevante Übersarungsflächen

Wie aus Abbildung 43 ersichtlich ist, werden die Übersarungen, welche auf ein Schadenpotenzial treffen, zuerst automatisch selektioniert. Die Selektion erfolgt pro Bearbeitungsregion (Abbildung 18).

Modell:	IN_Ueb (Toolbox: SiPro 93 IN_Ueb)
Input:	EV_Uebersarung_xxx, EV_Uebersarung_start_xxx und DA_GA.
Output:	IN_Ueb_start_xxx.



Modell 34: Relevante Übersarungen pro Gebiete.

Beschreibung des Modells:

- **EV_Uebersarung_aa: Make Feature Layer**
- **DA_GA: Make Feature Layer**
- **Select Layer By Location**: Input: EV_Uebersarung_aa_Layer, Selecting: DA_GA_Layer
- **Copy Feature**: Output: IN_Uebersarung_aa
- **EV_Uebersarung_start_aa: Make Feature Layer**
- **Join Field**: Input: EV_Uebersarung_start_aa_Layer, Input Join Field: ID, Join Table: IN_Uebersarung_aa, Output Join Field: ID
- **Copy Feature**: Output: EV_IN_Ueb_start_aa
- **EV_IN_Ueb_start_aa: Make Feature Layer**
- **Select Layer by Attribute**: ID_1 > 0
- **Copy Feature**: Output: IN_Ueb_start_aa

5.4.1.3 Bestimmung der relevanten Gerinne mit dem GIS-Tool "Utility Network Analyst"

Das GIS-Tool „Utility Network Analyst“ wurde semi-automatisch benutzt zur Bestimmung der relevanten Gerinne. In einem ersten Schritt wurde mit „select by location“ alle Flächen, die auf ein Schadenpotenzial treffen, selektioniert. In einem zweiten Schritt wurde diese Auswahl anhand einer Kartenanalyse verfeinert.

Als Eingangsdatensatz für diesen Schritt wurde nicht das Gerinnenetz aus dem Vector25-Datensatz der Swisstopo verwendet, sondern ein Netz, welches vom DTM25 abgeleitet wurde. Auf diese Weise konnte ein Gerinnenetz konstruiert werden, aus dem direkt die Fliessrichtung abgeleitet werden kann (Abbildung 46).

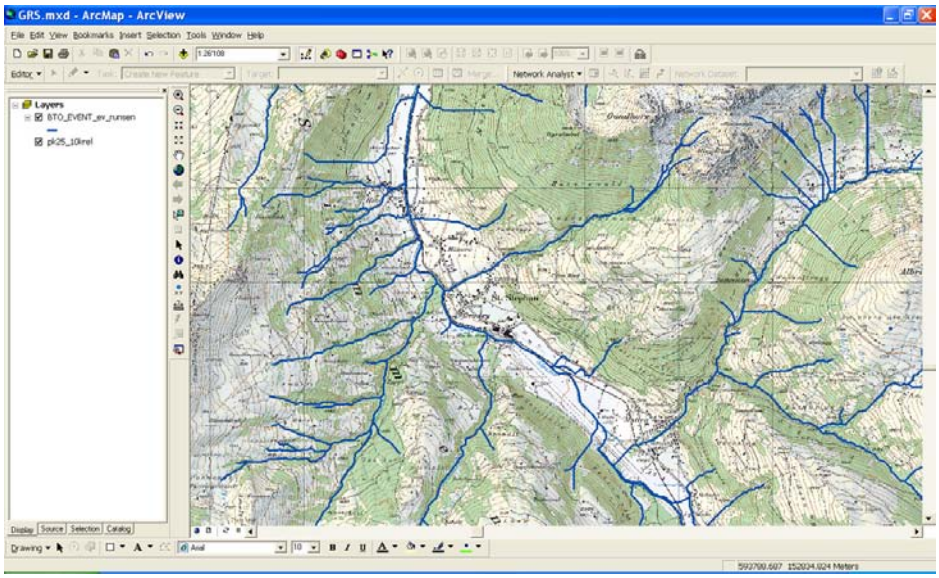


Abbildung 46: Aufgebautes Gerinnenetz mit der integrierten Fliessrichtung für das „Utility Network Analyst“ Tool.

Wie Abbildung 47 zeigt, wurden die Stellen, an welchen Geschiebe aus dem Gerinne austreten kann, violett markiert (Kapitel 5.4.1.2). Diese Gerinne können somit mit dem GIS-Tool „Utility Network Analyst“ selektioniert werden (grüne Punkte).

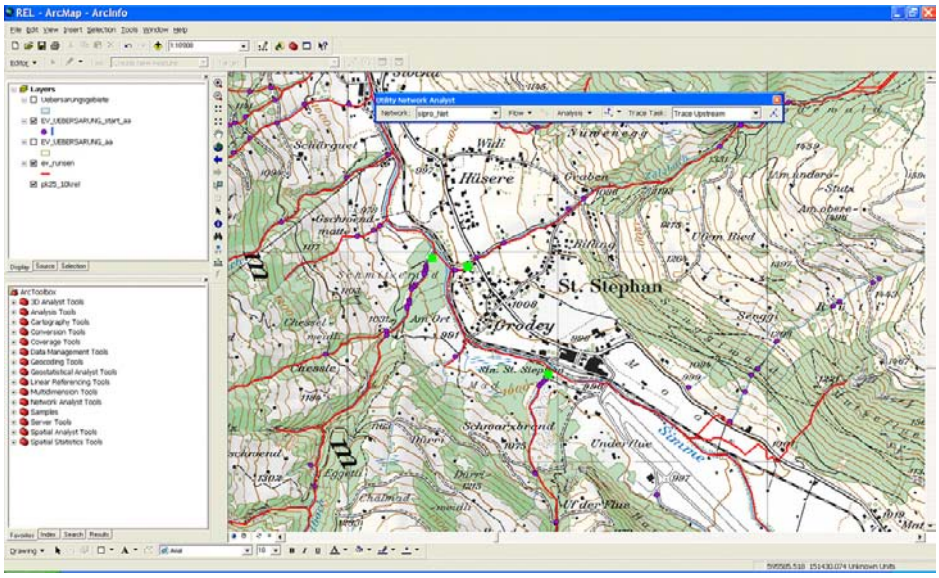


Abbildung 47: Festlegung der relevanten Gerinne.

Mit der Funktion „Trace_Upstream“ wird der obere Teil des Gerinnes ab den eingeführten Punkten automatisch gewählt. (Abbildung 48).

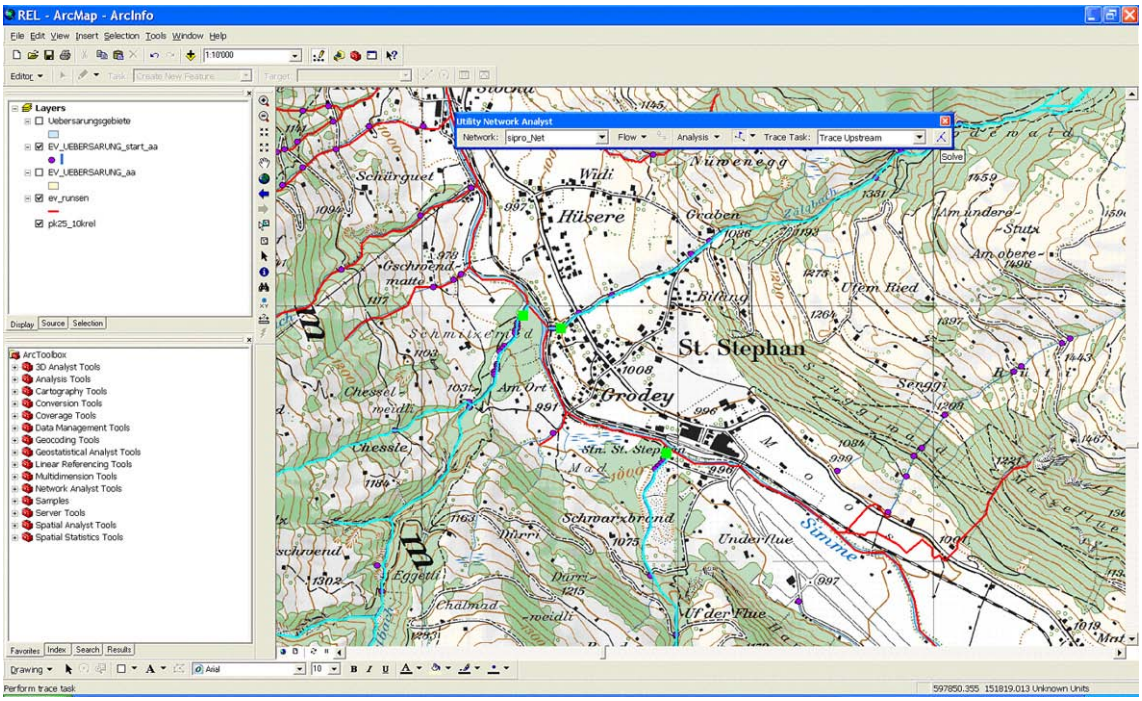


Abbildung 48: Mit der Funktion „Trace_Upstream“ wird das ganze Gerinne gewählt.

Damit dies funktioniert, müssen einige Optionen aktiviert sein (Abbildung 49). Auf diese Art kann das Objekt selektioniert und exportiert werden.

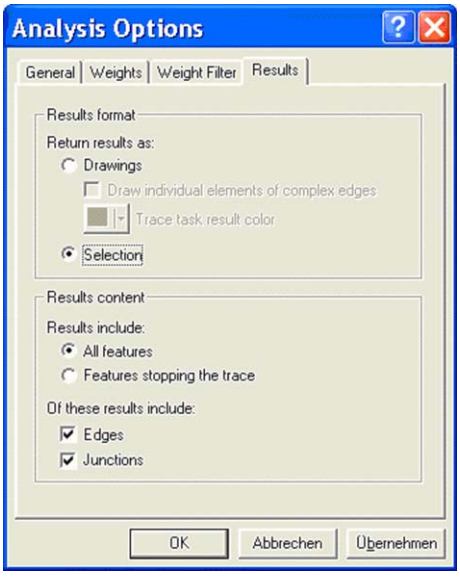


Abbildung 49: Einstellung bei der Optionen des Tools „Utility Network Analyst“.

Die gesamte Schweiz wurde auf diese Art visuell analysiert. Diese Arbeit musste sehr konservativ gemacht werden, weil ansonsten grosse Einzugsgebiete automatisch einbezogen worden wären: Wenn man beispielsweise einen Punkt im Berner Mattequartier wählen würde, würden sämtliche Gerinne im Berner Oberland automatisch selektioniert werden.

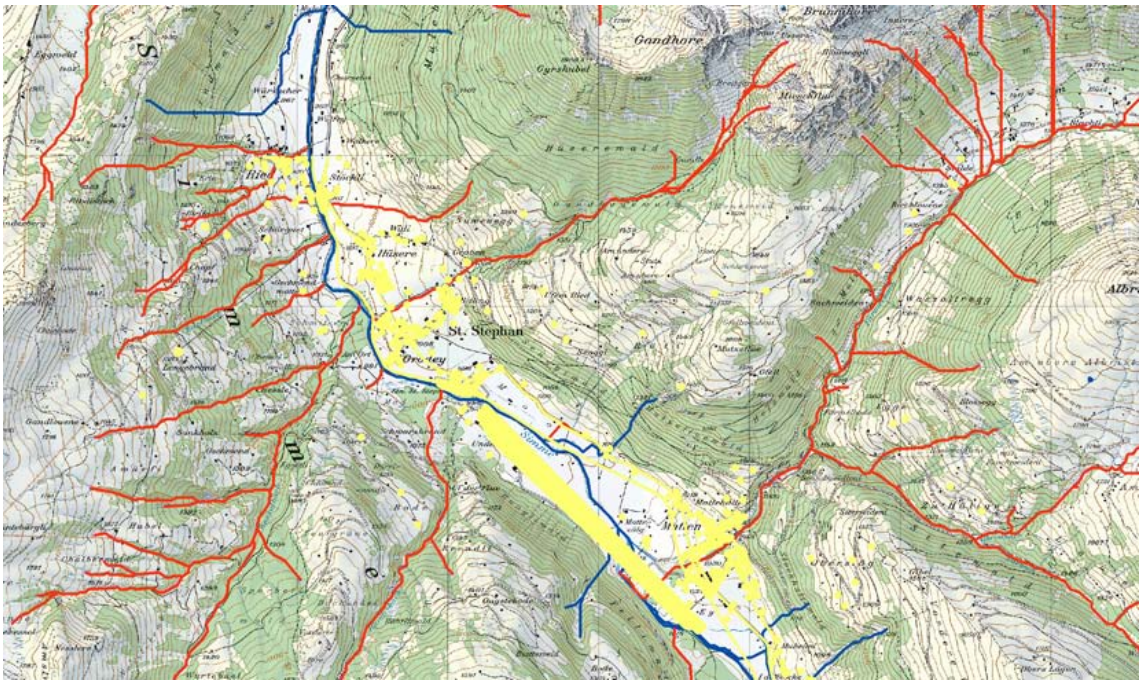


Abbildung 50: Relevante Gerinne (rot).

Das Resultat ist die Festlegung der relevanten Gerinnen für den Gefahrenprozess Übersarung (IN_RUNSE_UEB_CH).

5.4.2 Bestimmung der relevanten Gerinne mit Hilfe des Prozesses Murgang

5.4.2.1 Beschreibung der Modellierung

Die Modellierung des Prozesses “Murgang” ist im Kapitel 3.4 beschrieben. Sie wird zusammen mit dem Schadenpotenzial aus Kapitel 4 benutzt, um die relevanten Gerinne zu bestimmen. In einem ersten Schritt werden jene Murgangtrajektorien bestimmt, welche auf ein relevantes Schadenpotenzial zielen (Abbildung 51). Ausgehend von diesen Murgangtrajektorien werden im zweiten Schritt die relevanten Gerinne bestimmt (Abbildung 52).

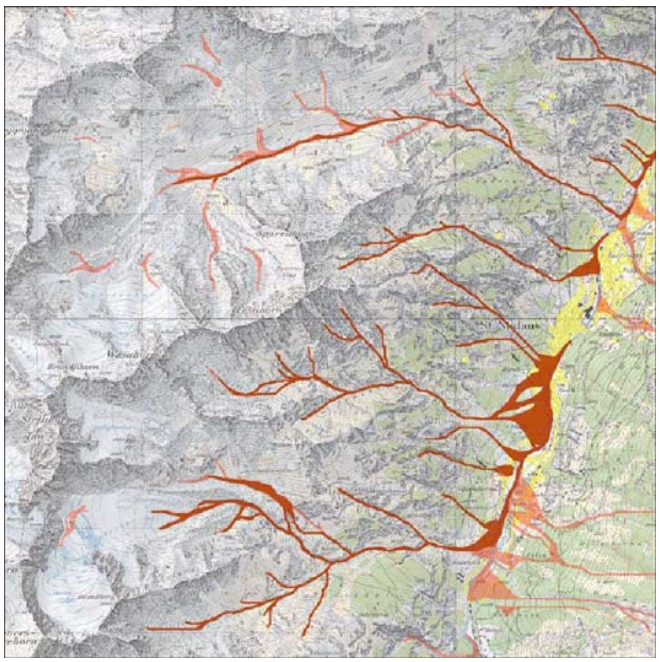


Abbildung 51: Schadenrelevante Murgangtrajektorien (dunkel rot).

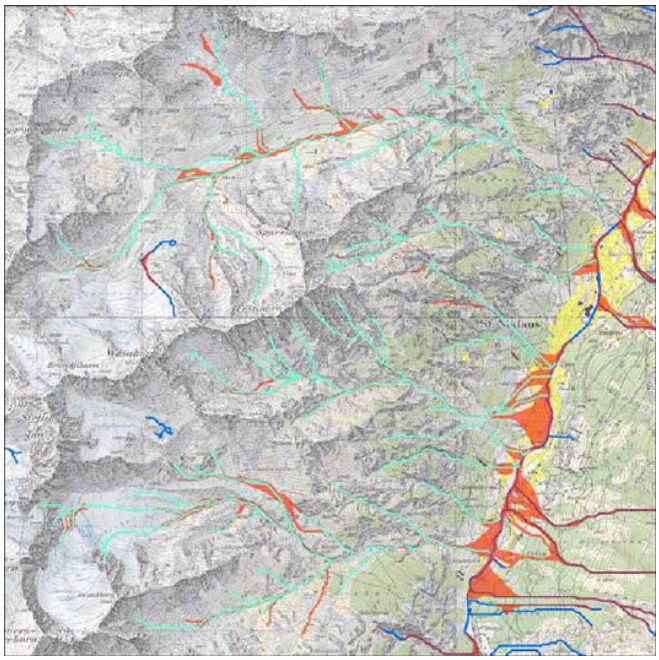


Abbildung 52: Auswahl der schadenrelevanten Gerinne (blau grün) aus dem Gerinnenetz (blau).

5.4.2.2 Bestimmung der relevanten Murgangtrajektorien für eine Kachel

Modell:	IN Mur 25 (Toolbox: SiPro 93 IN MUR)
Input:	EV_Mur_25, TILES10 und DA_GA_25
Output:	15 Shapes: 25ab, 25ac, 25ad, 25ba, 25bb, 25bc, 25bd, 25ca, 25cb, 25cc, 25cd, 25da, 25db, 25dc und 25dd.

Beschreibung des Modells

Dieses Modell berechnet alle relevanten Trajektorien der Kachel 25 gemäss Abbildung 16. Das Modell kann für alle Schadenpotenzialkategorien benutzt werden: Eisenbahnnetz, Anlagen, Gebäude, etc. Es wurde erstellt mit Hilfe von "List of value" und die folgenden Abbildungen zeigen die dabei benützten Kriterien.

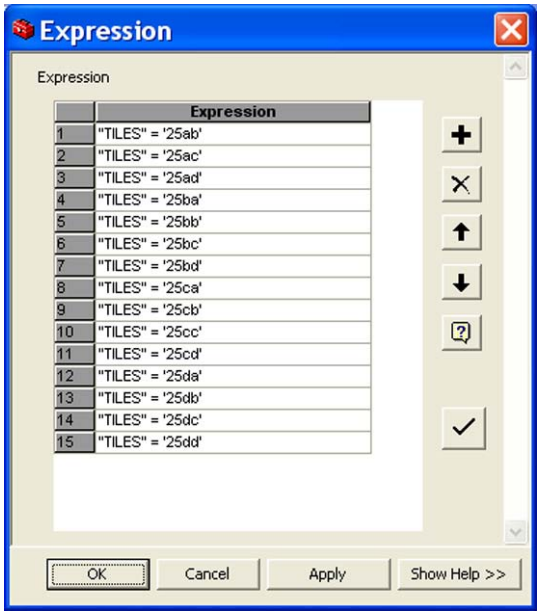


Abbildung 53: Expression für das Modell 35.

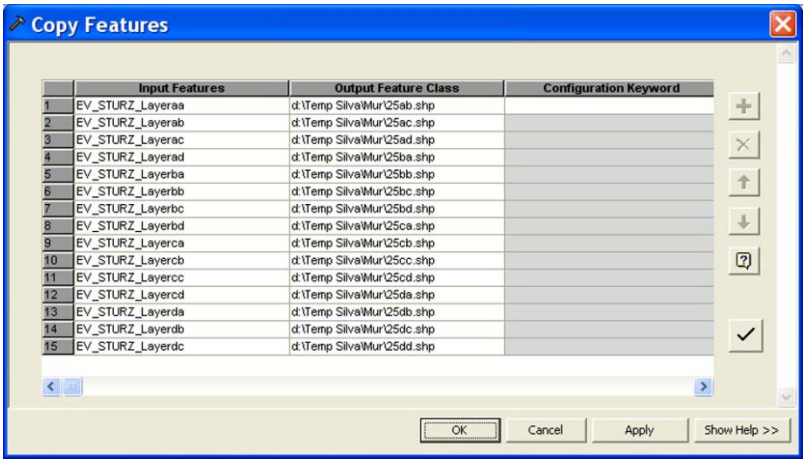
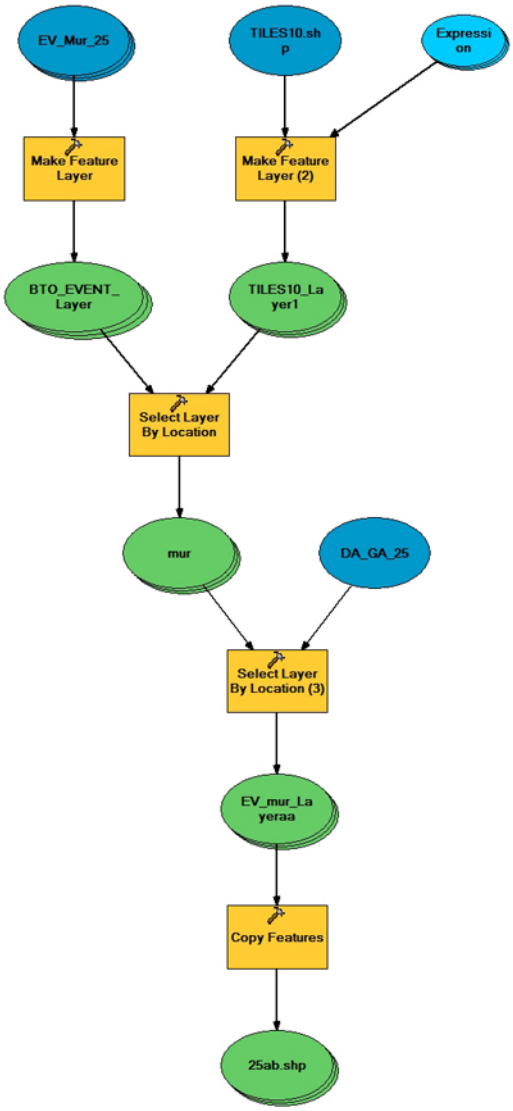


Abbildung 54: Resultate mit dem Modell 35.



Modell 35: Relevante Murgangtrajektorien.

Beschreibung des Modells:

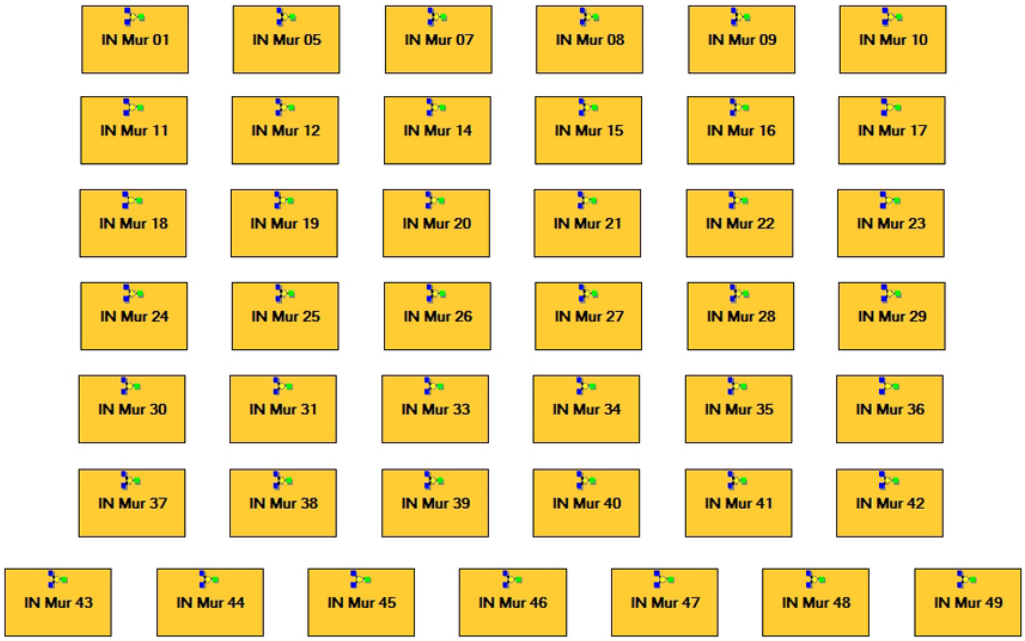
- **EV_Mur_25**: **Make Feature Layer**, Properties: List of values
- **TILES10**: **Make Feature Layer**, Expression: (Abbildung 53)
- **Select Layer By Location**: Input: EV_STURZ, Selecting: TILES10
- **Select Layer By Location**: Input: Sturz, Selecting: **DA_GA_07**, Selection type: SUB-SET_SELECTION
- **Copy Features**: (Abbildung 54)

5.4.2.3 Berechnung aller Kacheln zusammen

Modell:	IN Murgang (Toolbox: SiPro 93 IN MUR)
Input:	Modelle IN Mur 01, IN Mur 02, … bisIN Mur 49 EV_Mur_01 bis EV_Mur_49, TILES10 und DA_GA_01 bis DA_GA_49
Output:	ungefähr 500 Shapes: 1bc, 1bd, bis 49cc und 49cd.

Beschreibung des Modells

Für alle Kachel (Abbildung 16) wurde ein Modell analog zu Modell 35 erstellt. Die 49 einzelnen Modelle wurden danach in ein Gesamtmodell integriert. Somit kann die Berechnung selbständig ablaufen.



Modell 36: Relevante Murgangtrajektorien für die ganze Schweiz.

Zusätzliche GIS-Arbeiten:

Für alle Kacheln wurden die relevante Sturztrajektorien gepuffert und aufgelöst mit dem Modell 1: Umwandlung von Linien in Polygone. Danach wurden alle bearbeiteten Kacheln zusammengesetzt.

- **Dissolve: DR01** bis **DR49**
- **Merge**: Input: 1bc_diss, 1bd_diss, bis 49cc_diss und 49cd_diss
- **Dissolve:** Create Multipart Feature: non checked

In der Attributtabelle wurde eine neue Kolonne mit dem Name „MUR“ und dem Wert „1“ eingefügt.

- **Add Field**: Field Name: MUR, Field Type: SHORT
- **Calculate Field**: Field Name: MUR, Expression: 1
- **Delete Field**: alle unnötigen Felder werden gelöscht.

Wenn die Berechnung für alle Kacheln fertig ist, werden alle Shapes zusammengefügt (MERGE) und aufgelöst (DISSOLVE). **Das Schlussresultat ist das Shape IN_MUR_CH**.

5.4.2.4 Bestimmung der relevanten Gerinne

Nachdem die Murgangflächen definiert wurden (IN_MUR_CH), wurden sie mit dem Gerinnenetz (Runsen) aus Kapitel 5.4.1.3 überlagert (Abbildung 55:).

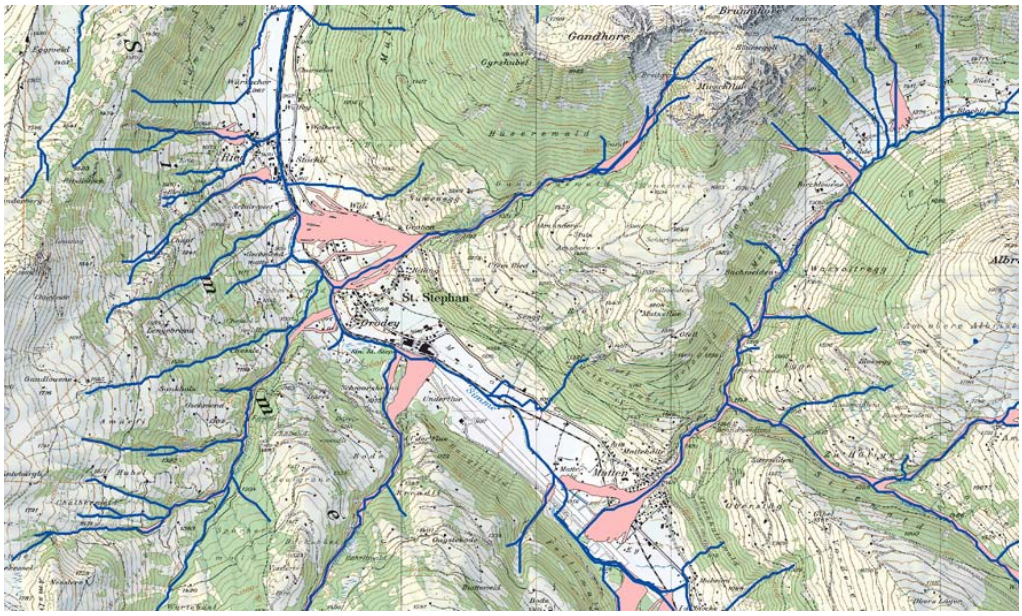


Abbildung 55: Überlagerung von Runsen (blau) und relevanten Murgangflächen (rosa).

Um schliesslich die relevanten Gerinne zu bestimmen, wurde das GIS-Tool „Utility Network Analyst“, analog zu Kapitel 5.4.1.3 verwendet. Dieses Mal wurden dazu aber nicht die Startpunkte der Übersarungsflächen sondern die Murgangflächen gebraucht.

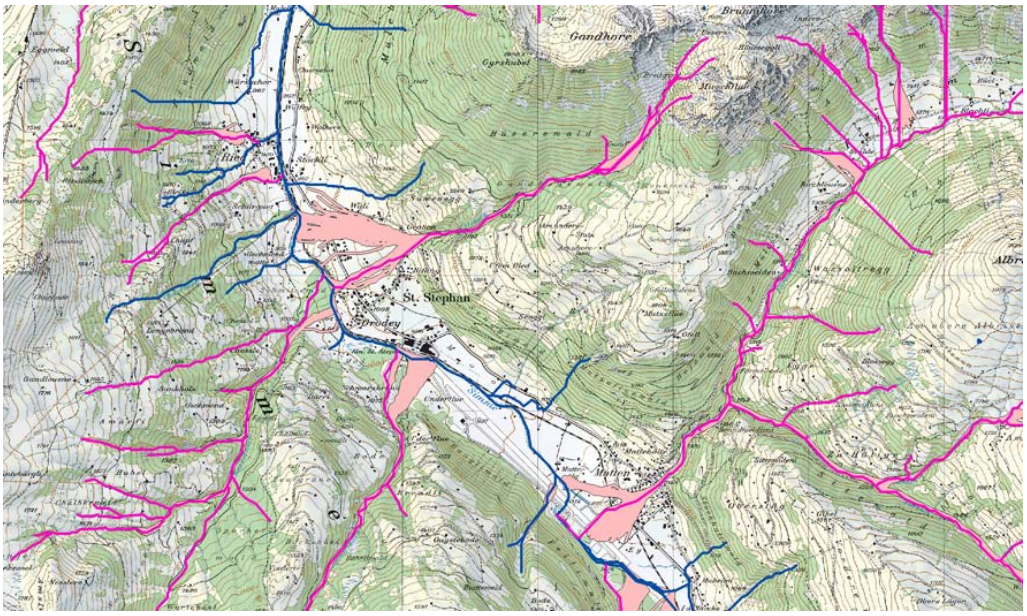


Abbildung 56: Relevante Runsen (violett) basierend auf den relevanten Murgangflächen (rosa).

Das Schlussresultat ist das Shape IN_RUNSE_MUR_CH.

5.4.3 Bestimmung der relevanten Gerinneprozesse

5.4.3.1 Beschreibung der Modellierung

Bevor die Gerinneprozesse modelliert werden konnten, mussten die beiden relevanten Gerinnenetze bezüglich Übersarung (**IN_RUNSE_UEB_CH**) und Murgang (**IN_RUNSE_MUR_CH**) mit dem GIS-Tool Merge zusammengeführt werden. Das Resultat wurde im Shapefile **IN_rGN_CH** gespeichert.

Die Modellierung der Gerinneprozesse lief grössenteils analog zur Modellierung der Prozesse Lawine (Kapitel 5.1), Sturz (Kapitel 5.2) und Hangmuren (Kapitel 5.3) ab. Einzig der Prozess Schwemmholz wurde mit einem anderen Ansatz modelliert. Aus diesem Grund sind hier bloss allfällige Unterschiede in der Modellierung der genannten Prozesse beschreiben.

5.4.3.2 Lawine: relevante Entstehungsgebiete

Im **ersten Schritt** wurden die relevanten Gerinne pro Lawinenregion bestimmt wie in Kapitel 5.1.2 beschreiben. Um keine weiteren GIS-Modelle erstellen zu müssen, wurde der Name des Shapefiles **IN_rGN_CH** in **DA_GA** geändert. Auf diese Weise konnte das Modell **DA_Lawine_pro_Gebiet** zur Bestimmung der relevanten Gerinne benutzt werden. Als Resultat wurden 30 Shapefiles generiert (**DA_adu** bis **DA_wil**).

Im **zweiten Schritt** wurden die relevanten Anrisszonen definiert (Kapitel 5.1.3).

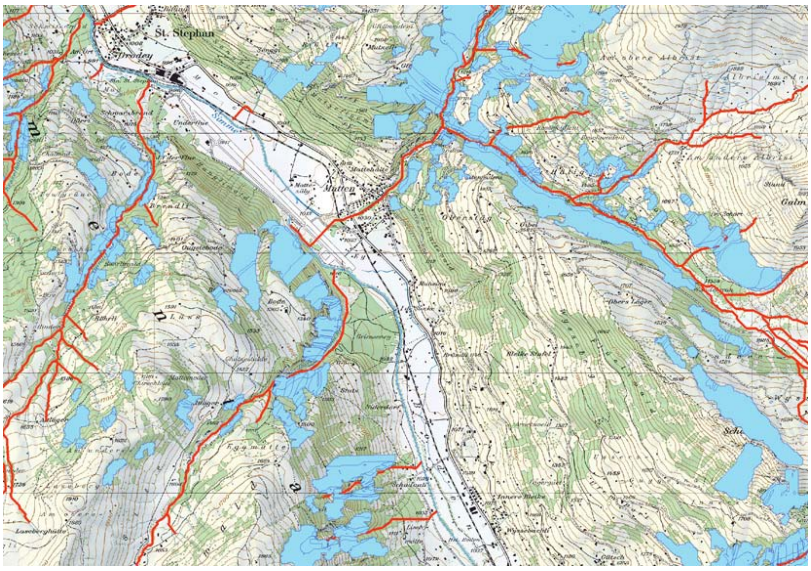


Abbildung 57: Relevante Anrissgebiete (blau) für die relevanten Gerinne (rot).

Andere GIS-Arbeiten, die nicht im Kapitel sind:

In der Attributtabelle wurde eine neue Kolonne mit dem Name „GRS_LAW“ und dem Wert „1“ eingefügt:

- **Add Field**: Field Name: GRS_LAW, Field Type: SHORT
- **Calculate Field**: Field Name: GRS_LAW, Expression: 1
- **Delete Field**: alle unnötigen Felder werden gelöscht.

Das Schlussresultat ist das Shape IN_GRS_LAWINE_CH.

5.4.3.3 Relevante Sturztrajektorien

Der Prozess entspricht der Beschreibung im Kapitel 5.2, einzig das Schadenpotenzial ist neu, nämlich das relevante Gerinnenetz aus dem vorhergehenden Kapitel. Um keine weiteren GIS-Modelle erstellen zu müssen, wurde der Name des Shapefiles **IN_rGN_CH** in **DA_GA** geändert. Auf diese Weise konnten die ursprünglichen Modelle verwendet werden.

Die Modellierung für alle Kachel mit 40 km Kantenlänge lief analog wie im Modell **IN_Sturz07** beschrieben und die Modellierung für die gesamte Schweiz entsprach dem Modell **IN_STURZ**.

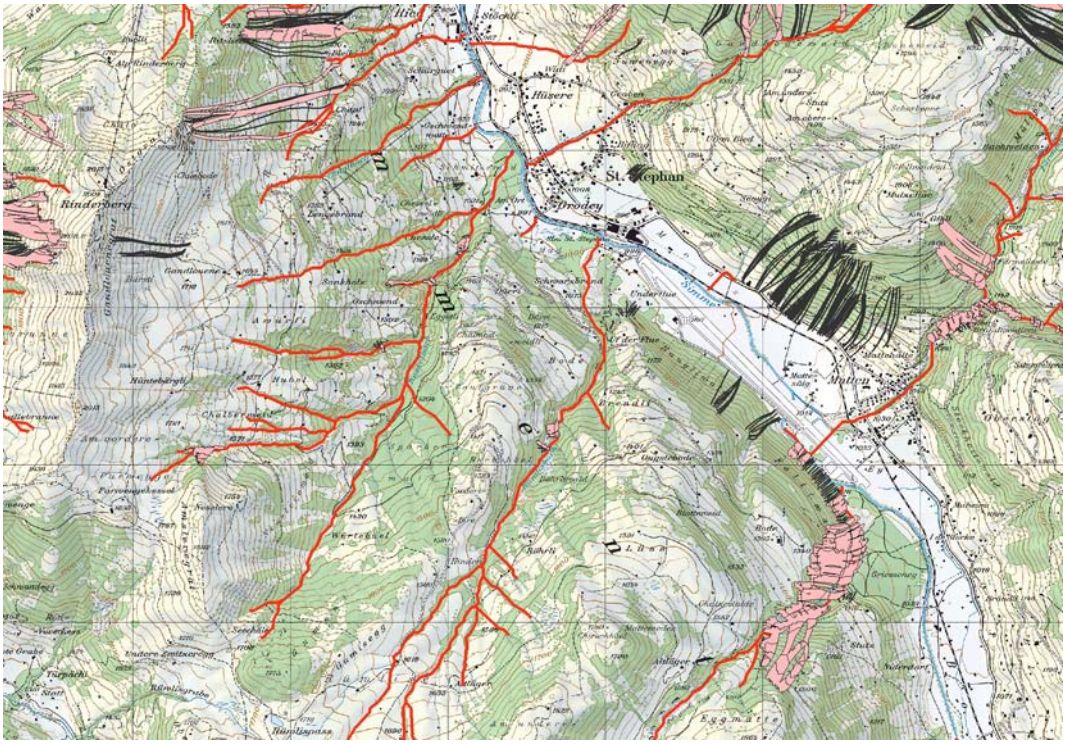


Abbildung 58: Relevante Sturzprozessflächen (rosa) für die relevanten Gerinne (rot). Die nicht relevanten Sturztrajektorien sind in schwarz.

Andere GIS-Arbeiten, die nicht im Kapitel sind:

In der Attributtabelle wurde eine neue Kolonne mit dem Name „GRS_STURZ“ und dem Wert „1“ eingefügt :

- **Add Field**: Field Name: GRS_ STURZ, Field Type: SHORT
- **Calculate Field**: Field Name: GRS_ STURZ, Expression: 1
- **Delete Field**: alle unnötigen Felder werden gelöscht.

Das Schlussresultat ist das Shape IN_GRS_STURZ_CH.

5.4.3.4 Relevante Hangmurentrajektorien

Der Prozess entspricht der Beschreibung im Kapitel 5.3, einzig das Schadenpotenzial ist neu, nämlich das relevante Gerinnenetz aus dem vorhergehenden Kapitel. Um keine weiteren GIS-Modelle erstellen zu müssen, wurde der Name des Shapefiles **IN_rGN_CH** in **DA_GA** geändert. Auf diese Weise konnten die ursprünglichen Modelle verwendet werden.

Die Modellierung für jede 40 km-Kachel lief analog wie im Modell **IN_HM01** beschrieben und die Modellierung für die gesamte Schweiz entsprach dem Modell **IN_HM**.

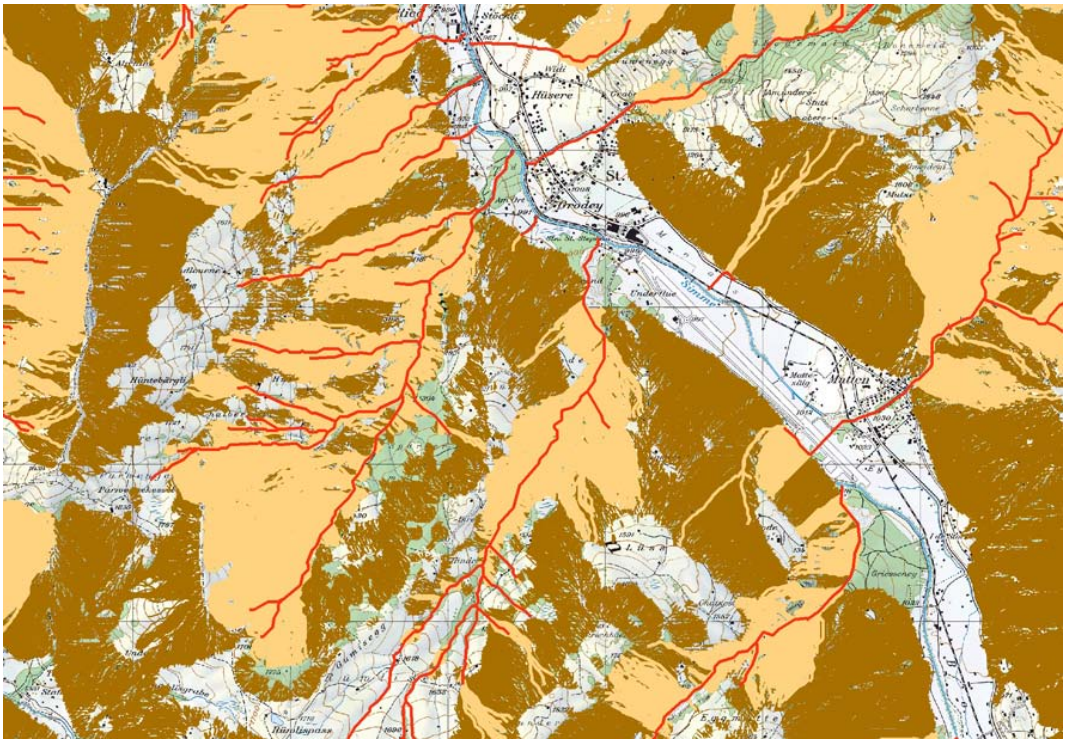


Abbildung 59: Relevante Hangmureflächen (beige) für die relevanten Gerinne (rot). Die nicht relevanten Hangmurentrajektorien sind in braun.

Andere GIS-Arbeiten, die nicht im Kapitel sind:

In der Attributtabelle wurde eine neue Kolonne mit dem Name „GRS_HM“ und dem Wert „1“ eingefügt :

- **Add Field**: Field Name: GRS_HM, Field Type: SHORT
- **Calculate Field**: Field Name: GRS_HM, Expression: 1
- **Delete Field**: alle unnötigen Felder werden gelöscht.

Das Schlussresultat ist das Shape IN_GRS_HM_CH.

5.4.3.5 Schwemmholz

Mobilisierung von Schwemmholz und Ufererosion können reduziert werden, wenn die Waldbestockung an den Gerinneeinhängen gut unterhalten ist.

Aus diesem Grund wurde beidseitig zum relevanten Gerinne ein Puffer von 50m ausgeschieden. Der in dieser relevanten Schwemmholzfläche stockende Wald gilt als Schutzwald.

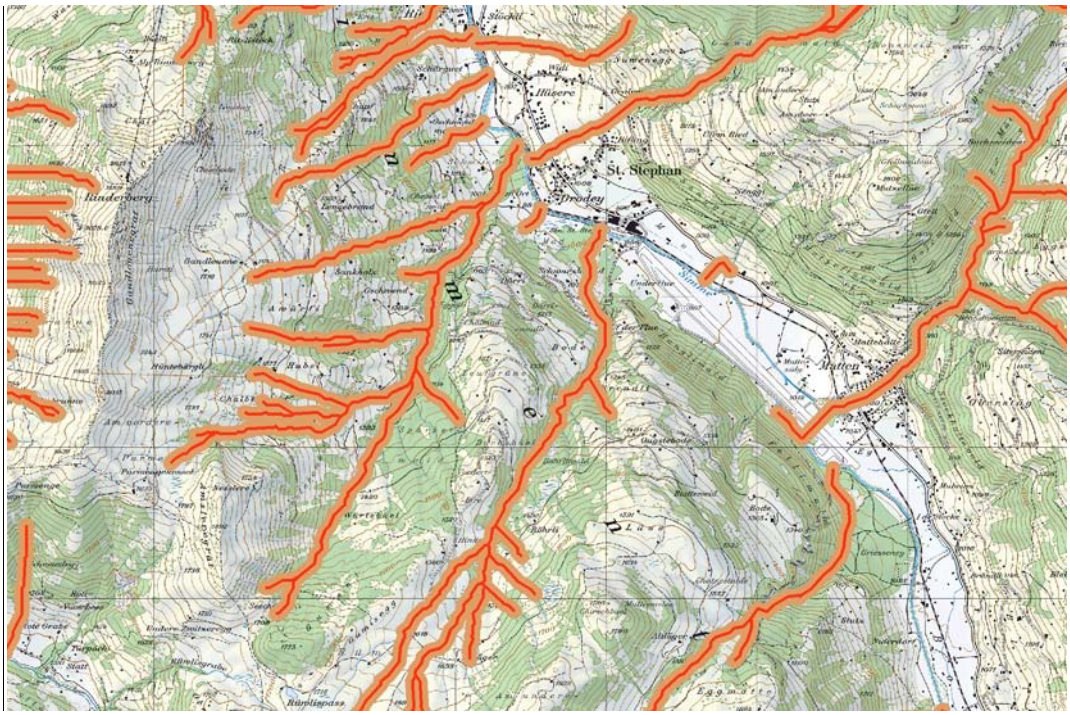


Abbildung 60: Relevante Schwemmholzflächen (beige). Rot sind die relevanten Gerinne.

GIS-Arbeiten:

- **Buffer**: Input: IN_rGN_CH, Linear Unit 50 Meters
- **Dissolve:** Create Multipart Feature: non checked

In der Attributtabelle wurde eine neue Kolonne mit dem Name „GRS_HM“ und dem Wert „1“ eingefügt :

- **Add Field**: Field Name: GRS_SchH, Field Type: SHORT
- **Calculate Field**: Field Name: GRS_SchH, Expression: 1
- **Delete Field**: alle unnötigen Felder werden gelöscht.

Das Schlussresultat ist das Shape IN_GRS_SchHolz_CH.

5.4.3.6 Zusammenfassung der relevanten Gerinneprozesse

Nachdem alle relevante Prozesse (Lawine, Sturz, Hangmure und Schwemmholz), die auf ein relevantes Gerinne treffen können, berechnet worden waren, wurden diese in einem Shapefile zusammengefügt.

GIS- Arbeiten:

- **Union**: Input: IN_GRS_LAWINE_CH, IN_GRS_STURZ_CH, IN_GRS_HM_CH und IN_GRS_SchHolz_CH.
- **Delete Field**: alle unnötigen Felder wurden gelöscht. Übrig blieben nur die Felder: GRS_LAW, GRS_STURZ, GRS_HM und GRS_SchH.

Das Schlussresultat ist IN_GRS_Detail_CH. Aus diesem Shapefile ist für jedes relevante Gerinne ersichtlich, welcher Prozess auf dieses Gerinne trifft.

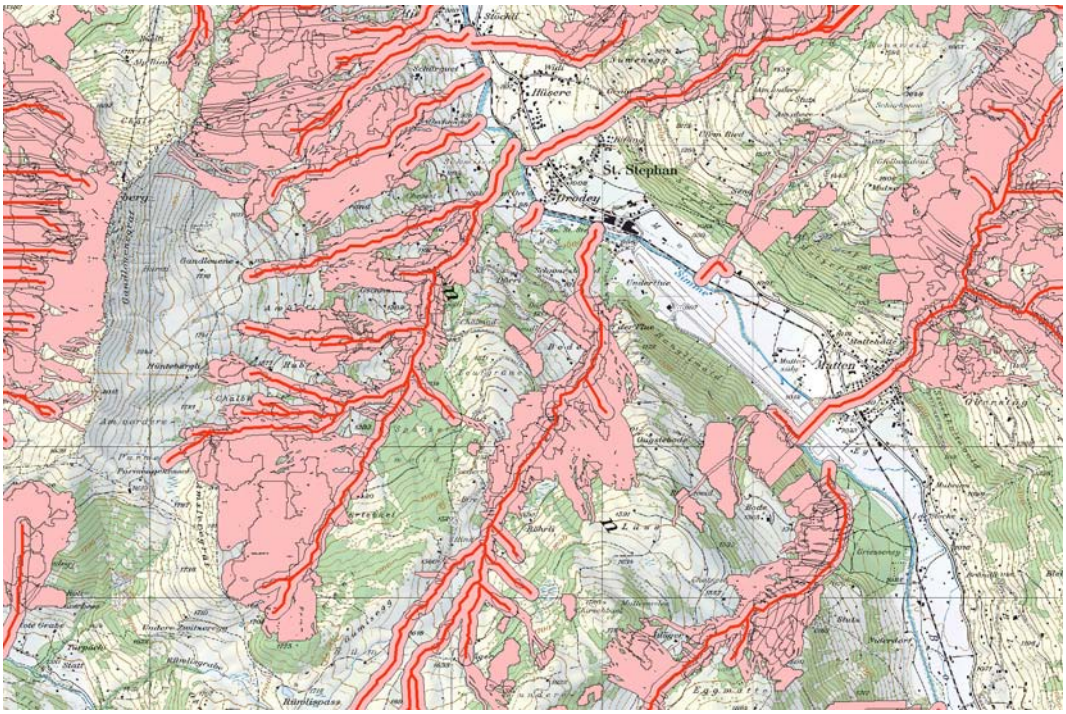


Abbildung 61: Detaillierte relevante Prozessflächen (rosa), die die relevanten Gerinne (rot) treffen.

In der Attributtabelle kann man die relevanten Naturgefahrenprozesse zurückverfolgen.

Attributes of IN_GRS_Detail_2010

FID	Shape	GRS_Sturz	GRS_SchH	GRS_Law	GRS_HM
0	Polygon	1	0	0	0
1	Polygon	1	0	0	0
2	Polygon	1	0	0	0
3	Polygon	1	0	0	0
4	Polygon	1	0	0	0
5	Polygon	1	0	0	0
6	Polygon	1	0	0	0
7	Polygon	1	0	0	0
8	Polygon	1	0	0	0
9	Polygon	1	0	0	0
10	Polygon	1	0	0	0
11	Polygon	1	0	0	0
12	Polygon	1	0	0	0
13	Polygon	1	0	0	0
14	Polygon	1	0	0	0
15	Polygon	1	0	0	0
16	Polygon	1	0	0	0
17	Polygon	1	0	0	0
18	Polygon	1	0	0	0
19	Polygon	1	0	0	0
20	Polygon	1	0	0	0
21	Polygon	1	0	0	0
22	Polygon	1	0	0	0
23	Polygon	1	0	0	0

Record: 1 Show: All Selected Records

Abbildung 62: Detaillierte relevante Gefahrprozessfläche für die relevanten Gerinne.

Schliesslich wurden die Flächen in diesem Shapefile noch zusammengefasst (Dissolve), damit man einen einzigen Layer mit allen für die Gerinne relevanten Prozessen hat. Das Shape enthält nur noch die Information zu den Gerinneprozesse, ohne weitere Aufteilung auf die darunterliegenden Gefahrenprozesse.

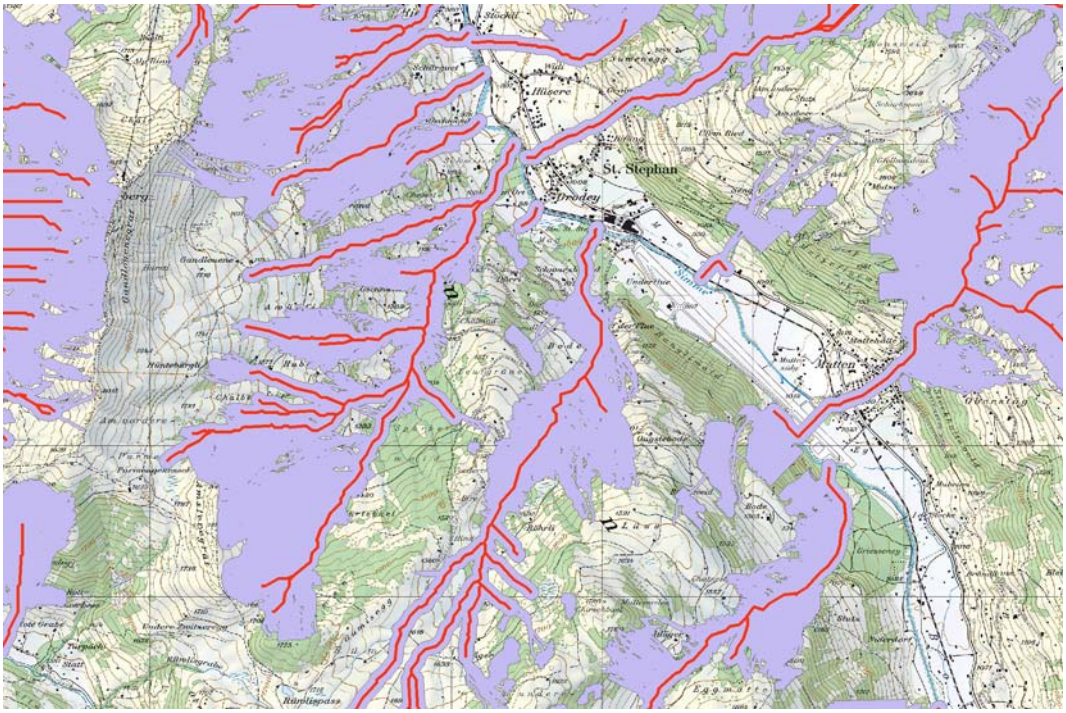


Abbildung 63: Relevante Prozessflächen (violett), die die relevanten Gerinne (rot) treffen.

GIS-Arbeiten:

- **Dissolve:** Input: IN_GRS_Detail_CH, Create Multipart Feature: non checked

In der Attributtabelle wurde eine neue Kolonne mit dem Name „GRS_HM“ und dem Wert „1“ eingefügt:

- **Add Field**: Field Name: GRS, Field Type: SHORT
- **Calculate Field**: Field Name: GRS, Expression: 1
- **Delete Field**: alle unnötigen Felder werden gelöscht.

Das Schlussresultat ist das Shape IN_GRS_CH

5.5 Zusammenfassung den relevanten Prozessen

Aus den Teilresultaten der modellierten relevanten Prozesse aus den Kapiteln 5.1 bis 5.4 (Lawine, Sturz, Hangmure und Gerinneprozesse), wurde ein Shapefile erstellt.

GIS-Arbeitn:

- **Union**: Input: IN_LAWINE_CH, IN_STURZ_CH, IN_HM_CH und IN_GRS_CH.
- **Delete Field**: alle unnötigen Felder wurden gelöscht. Übrig bleiben die Felder: LAW, STURZ, HM und GRS.

Das Schlussresultat ist IN_Prozesse_CH. Mit Hilfe dieses Shapefiles kann man nun ableiten, welcher Prozess (Lawine, Sturz, Hangmure und Gerinneprozesse) auf ein Schadenpotenzial trifft (Abbildung 64).

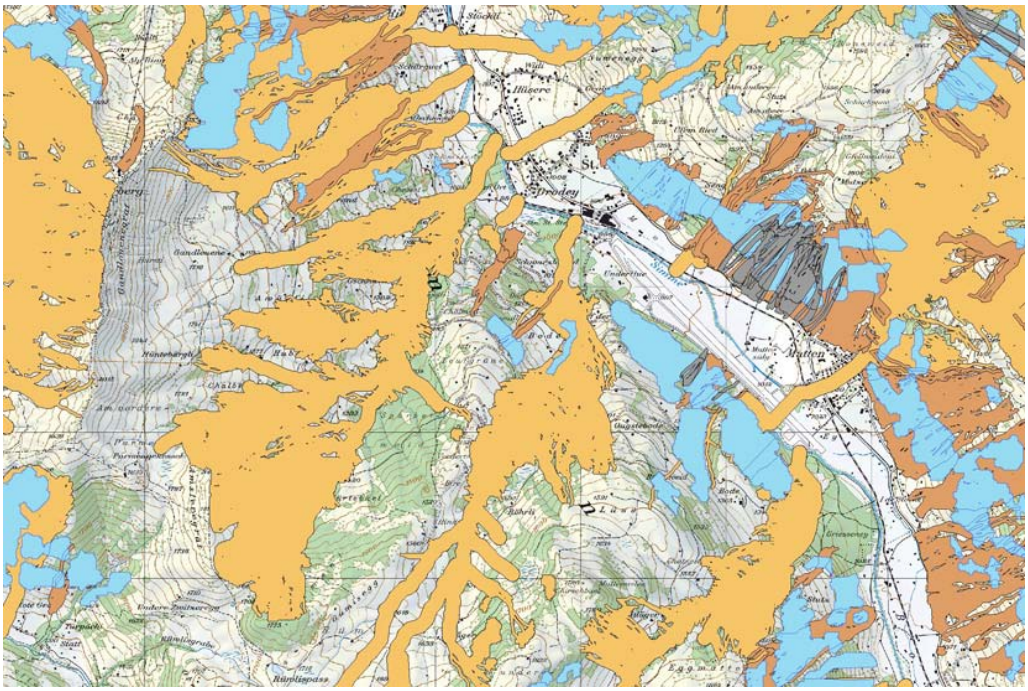
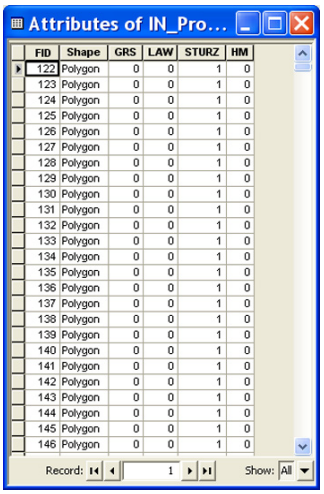


Abbildung 64: Relevante Prozessflächen aus SilvaProtect-CH: Lawine (blau), Sturz (grau), Hangmure (braun), Gerinneprozesse (beige)

In der Attribut-Tabelle kann man die relevanten Naturgefahrenprozesse wiederfinden.



Attributes of IN_Pro...

FID	Shape	GRS	LAW	STURZ	HM
122	Polygon	0	0	1	0
123	Polygon	0	0	1	0
124	Polygon	0	0	1	0
125	Polygon	0	0	1	0
126	Polygon	0	0	1	0
127	Polygon	0	0	1	0
128	Polygon	0	0	1	0
129	Polygon	0	0	1	0
130	Polygon	0	0	1	0
131	Polygon	0	0	1	0
132	Polygon	0	0	1	0
133	Polygon	0	0	1	0
134	Polygon	0	0	1	0
135	Polygon	0	0	1	0
136	Polygon	0	0	1	0
137	Polygon	0	0	1	0
138	Polygon	0	0	1	0
139	Polygon	0	0	1	0
140	Polygon	0	0	1	0
141	Polygon	0	0	1	0
142	Polygon	0	0	1	0
143	Polygon	0	0	1	0
144	Polygon	0	0	1	0
145	Polygon	0	0	1	0
146	Polygon	0	0	1	0

Record: 1 Show: All

Abbildung 65: Attribut-Tabelle mit den detaillierten Prozessen.

6 Waldfläche (SILVA)

Um die relevanten Prozessflächen im Wald zu modellieren, braucht es einerseits die relevanten Prozessflächen und anderseits die Waldfläche. Erstere stammt aus Kapitel 5 Relevante Prozesse (INTERSECT), letzere wird im hier beschreibenen Modul bereitgestellt.

Die homogensten Daten für die Erstellung der Waldfläche kommen aus den Datensatz „Vektor 25“. Zu diesem Datensatz mussten zusätzlich noch die Vivian- und Lotharflächen, welche teilweise in Vektor 25 fehlen, digitalisiert werden.

Aus den gennanten Eingangsdaten wurden folgende Objekte verwendet:

- Vector25 Primärflächen
    - Wald (geschlossen)
    - Wald offen
    - Geröll in Wald
    - Geröll in offenem Wald
    - Sumpf in Wald
    - Sumpf in offenem Wald
- Vivian90: Alle Flächen
- Lothar99: Alle Flächen

Obwohl für gewisse Prozesse (insbesondere Lawinenanrisse) auch die Gebüsche eine Schutzfunktion haben können, wurde das Objekt „Gebüsch“ aus dem Datensatz „Vector25 Primärflächen“ nicht verwendet, da es in sich nicht weiter differenzierbar ist.

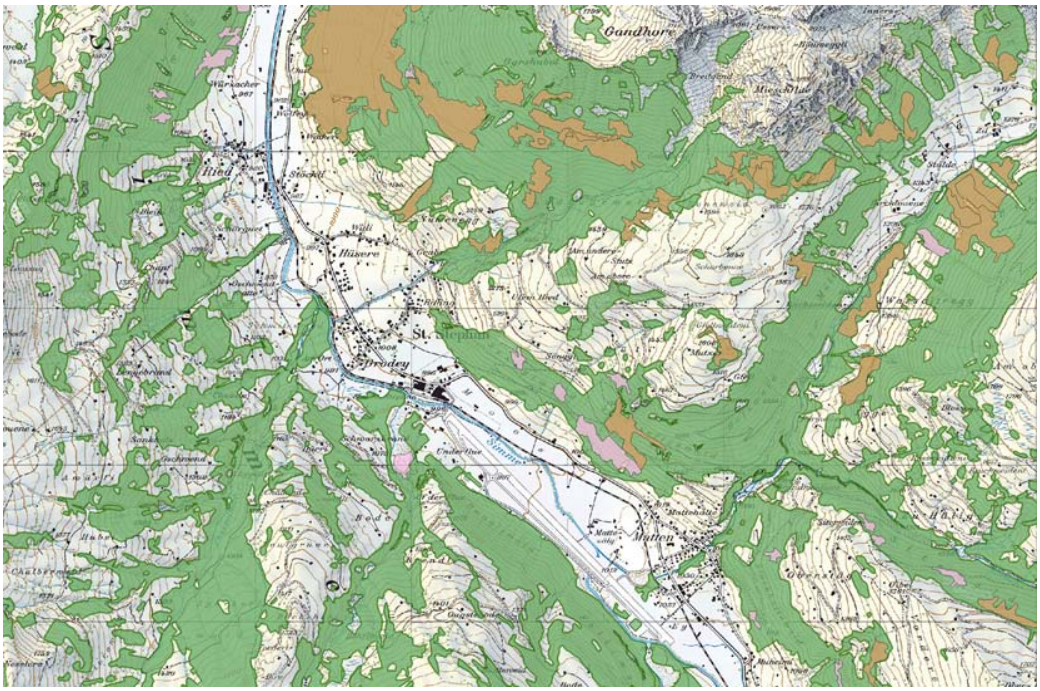
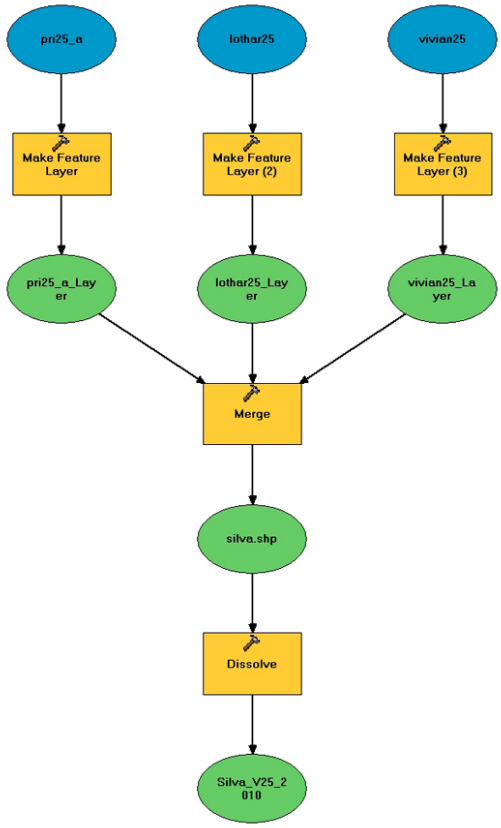


Abbildung 66: Waldflächen aus dem Vector25 (grün), der Vivianflächen (violett) und der Lotharflächen (braun).

Modell: Silva (Toolbox: SiPro 93 Silva)

Input: pri25_a, lothar25 und vivian25

Output: Silva_V25.



Modell 37: Bereitstellung der Waldfläche mit Hilfe von Vector25, Lothar- und Vivianflächen.

Beschreibung des Modells:

Vector25 Primärflächen

- **pri25_a**: **Make Feature Layer**: "OBJECTVAL" = 'Z_GerWa' OR "OBJECTVAL" = 'Z_GerWaO' OR "OBJECTVAL" = 'Z_SumWa' OR "OBJECTVAL" = 'Z_SumWaO' OR "OBJECTVAL" = 'Z_Wald' OR "OBJECTVAL" = 'Z_WaldOf'

Vivian90

- **Vivian25**: **Make Feature Layer**

Lothar99

- **Lothar25**: **Make Feature Layer**

- **Merge**: Input: pri25_a_Layer, lothar25_Layer und vivian25_Layer
- **Dissolve**: Create Multipart Feature: non checked

7 Schutzwald- und Schadenpotenzialindex (SYNTHESE)

Im Modul SYNTHESE wurden die eigentlichen Endresultate erarbeitet:

- durch den Verschnitt der schadenrelevanten Prozessflächen mit der Waldfläche (SILVA) wurde **der Schutzwaldindex berechnet;**
- durch den Verschnitt der Prozessflächen mit den Schadenpotenzialobjekten (DAMAGE) wurde **der Schadenpotenzialindex berechnet.**

7.1 Schutzwaldindex

Beschreibung und Resultate des Modellierung

Der Verschnitt zwischen den schadenrelevanten Prozessflächen und der Waldfläche ergibt als Resultat die schadenrelevanten Prozessflächen im Wald. Diese Berechnung läuft pro Kanton. Wegen der grossen Datenmenge ist es für die Workstation nicht möglich, die ganze Schweiz gleichzeitig zu berechnen.

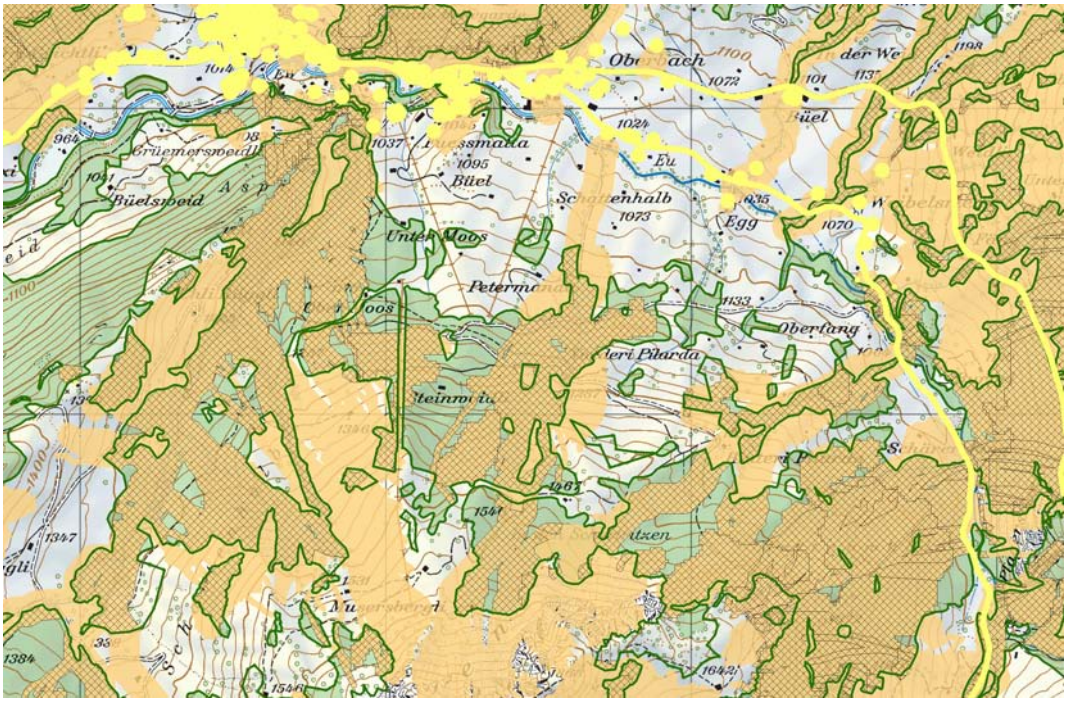
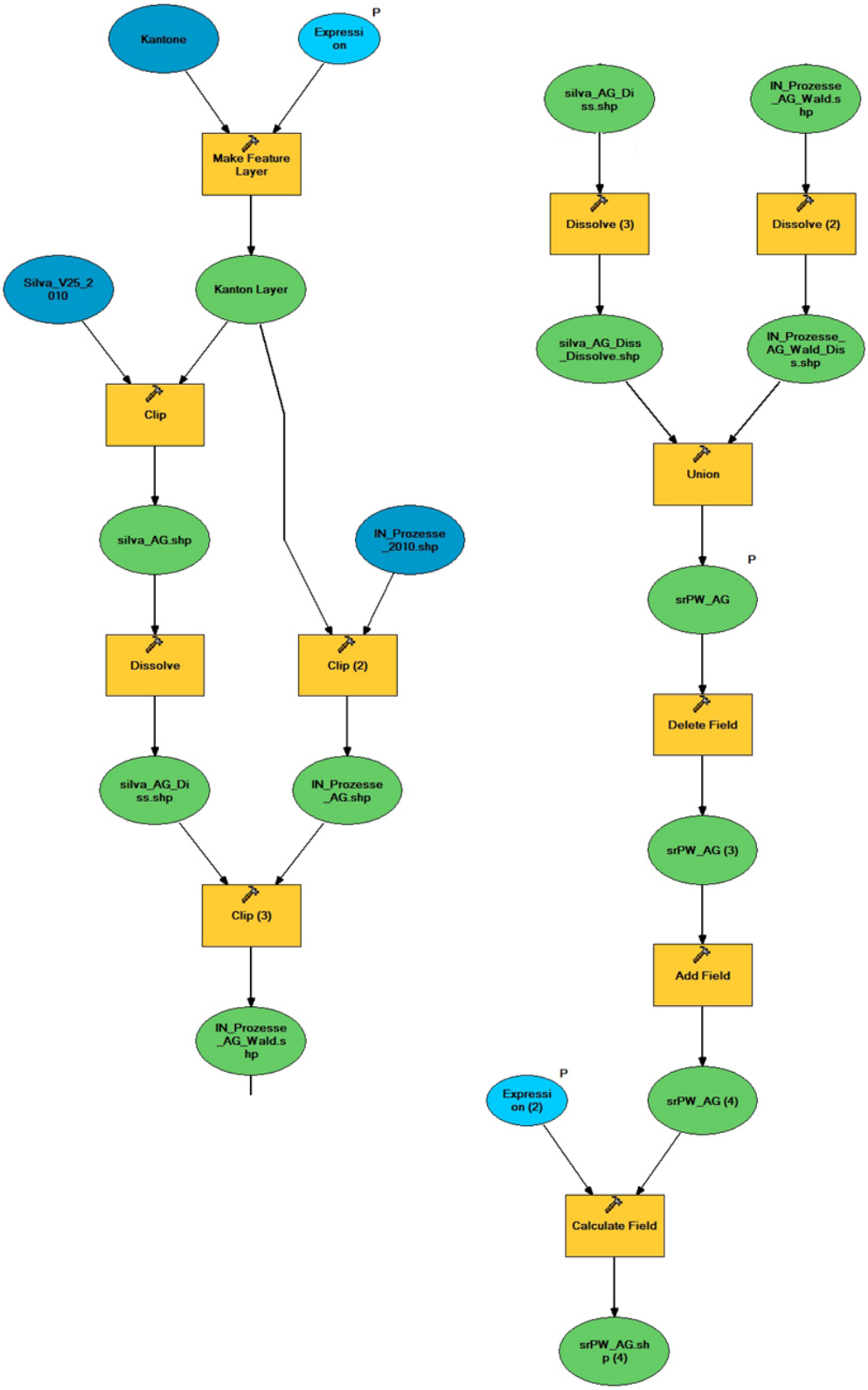


Abbildung 67: Relevante Prozessflächen im Wald (schwarz schraffiert);
Schadenpotenzial (gelb), relevante Prozesse (beige) und Wald (grün).

Modell:	Schutzwaldindex (Toolbox: SiPro 93 SYNTHESE)
Input:	Kantone, Silva_V25 und IN_Prozesse_CH.
Output:	srPW_AG bis srPW_ZH.



Modell 38: Berechnung des Schutzwaldindex.

Beschreibung des Modells:

- **Kantone**: **Make Feature Layer**: **Expression**: (Model Parameter); "KT" = 'AG' Abbildung 68: Expression).

Waldfläche pro Kanton

- **Clip**: Input: **Silva_V25_2010**; Clip: Kanton Layer
- **Dissolve**: Input: Silva_AG; Create Multipart Feature: non checked
- **Dissolve (3)**: Input: Silva_AG_Diss; Create Multipart Feature: checked

Relevante Prozessflächen pro Kanton

- **Clip (2)**: Input: **IN_Prozesse_2010**; Clip: Kanton Layer

Relevante Prozessflächen im Wald

- **Clip (3)**: Input: IN_Prozesse_AG; Clip: Silva_AG_Diss
- **Dissolve (2)**: Input: IN_Prozesse_AG_Wald; Dissolve Field(s): GRS, LAW, STURZ und HM: checked; Create Multipart Feature: checked
- **Union**: Input Features: IN_Prozesse_AG_Wald_diss und Silva_AG_Diss_Dissolve
- **Delete Field**: alle unnötige Kolonne sind damit gelöscht.
- **Add Field**: Input Table: srPW_AG; Field Name: ‚KT'; Field Type: ‚TEXT
- **Calculate Field**: Input Table: srPW_AG; Field Name: ‚KT'; Expression (Abbildung 68: Expression 2).
- **Expression**: Expression = 'AG'

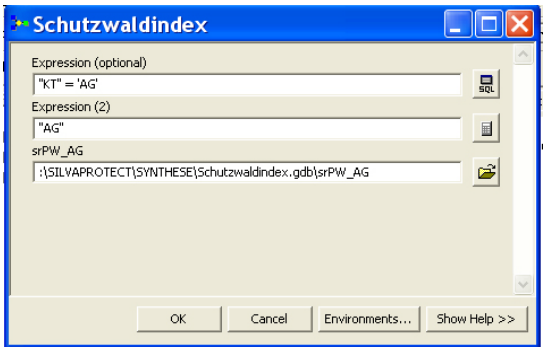
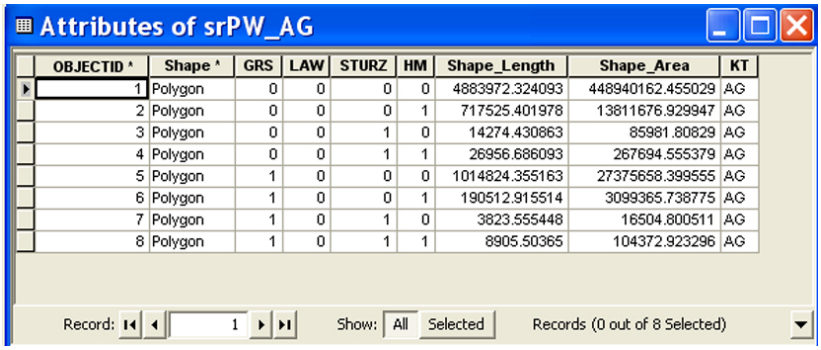


Abbildung 68: Nötige Informationen für die Berechnung des Schutzwaldindex.

Das Schlussresultat liegt in einem Shapefile vor. Seine Attribut-Tabelle (Abbildung 69) informiert über die Prozesse, die in einem Kanton vorhanden sind, sowie über das Ausmass an Fläche, das von diesen Prozessen betroffen ist.



OBJECTID *	Shape *	GRS	LAW	STURZ	HM	Shape_Length	Shape_Area	KT
1	Polygon	0	0	0	0	4883972.324093	448940162.455029	AG
2	Polygon	0	0	0	1	717525.401978	13811676.929947	AG
3	Polygon	0	0	1	0	14274.430863	85981.80829	AG
4	Polygon	0	0	1	1	26956.686093	267694.555379	AG
5	Polygon	1	0	0	0	1014824.355163	27375658.399555	AG
6	Polygon	1	0	0	1	190512.915514	3099365.738775	AG
7	Polygon	1	0	1	0	3823.555448	16504.800511	AG
8	Polygon	1	0	1	1	8905.50365	104372.923296	AG

Abbildung 69: Attributen der schadenrelevanten Prozessflächen im Wald eines Kantons.

Diese Berechnung wurde für alle Kantone gemacht. Am Ende wurden alle Shapefiles in einem Shapefile (srPW_CH) zusammengefasst (MERGE) und die neue Attribut-Tabelle in ein EXCEL-Format exportiert. Danach konnte der Schutzwaldindex direkt mit Hilfe einer Pivot-Tabelle berechnet werden.

7.2 Schadenpotenzialindex

Für die Berechnung des Schadenpotenzialindexes wurden nur jene Prozesse berücksichtigt, die einen direkten Bezug zum Schadenpotenzial haben (Lawine, Murgang, Übersarung und Sturz). Beim Schadenpotenzialindex wurde zudem eine Gewichtung vorgenommen (Abbildung 70).

Die Modellierung wurde dabei pro Kanton vorgenommen, da die grosse Datenmenge keine Berechnung über die ganze Schweiz erlaubte.

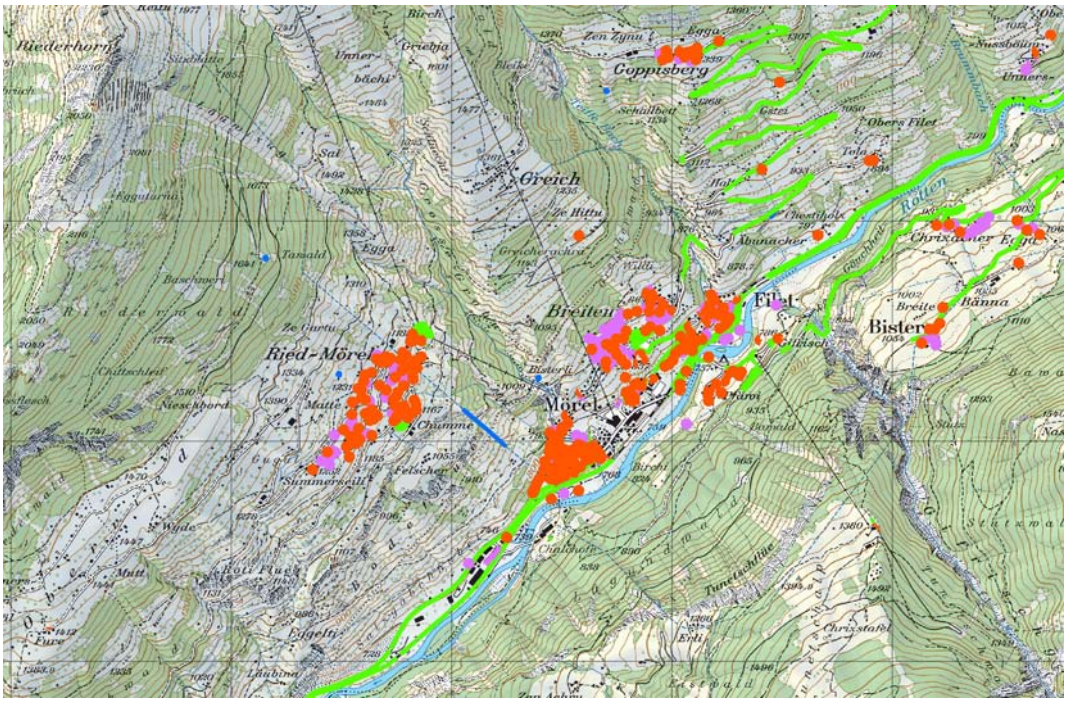
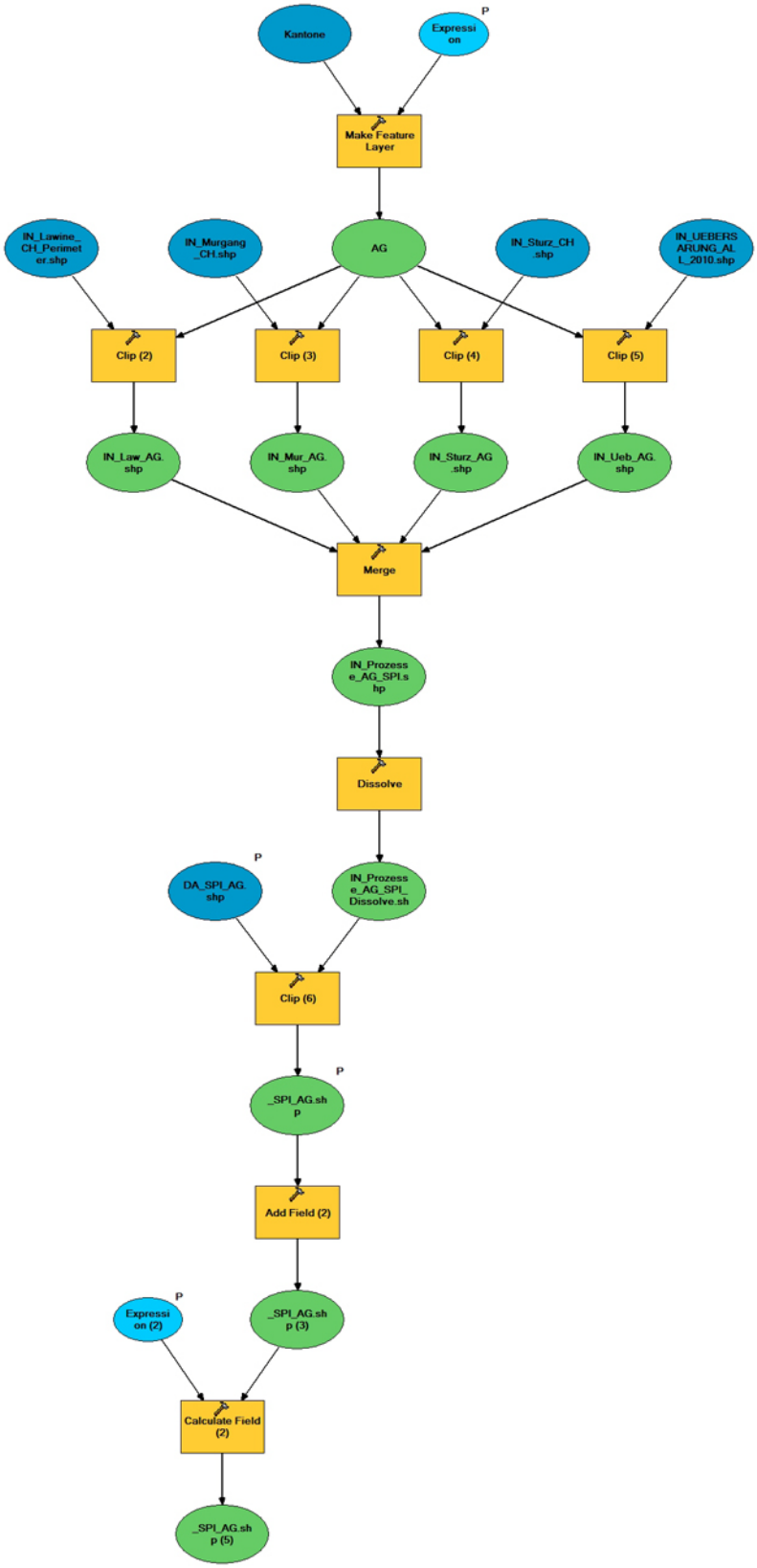


Abbildung 70: Betroffenes Schadenpotenzial: Gewichtung 10 (rot), 5 (violett), 3 (grün) und 1 (blau).

Modell:	Schadenpotenzialindex (Toolbox: SiPro 93 SYNTHESE)
Input:	Kantone, IN_Lawine_CH, IN_Murgang_CH, IN_Sturz_CH und IN_Uebersarung_CH..
Output:	SPI_AG bis SPI_ZH.



Modell 39: Schadenpotenzialindex pro Kanton.

Beschreibung des Modells:

- **Kantone**: **Make Feature Layer**: **Expression**: (Model Parameter); "KT" = 'AG' (Abbildung 71: Expression).
- **Clip (2)**: Input: **IN_Lawine_CH**; Clip: Kanton Layer
- **Clip (3)**: Input: **IN_Murgang_CH**; Clip: Kanton Layer
- **Clip (4)**: Input: **IN_Sturz_CH**; Clip: Kanton Layer
- **Clip (5)**: Input: **IN_Uebersarung_CH**; Clip: Kanton Layer
- **Merge**: Input: IN_Law_AG, IN_Mur_AG, IN_Sturz_AG und IN_Ueb_AG
- **Dissolve**: Input: IN_Prozesse_AG; Create Multipart Feature: non checked
- **Clip (6)**: Input: **DA_SPI_AG**; Clip: IN_Prozesse_AG
- **Add Field**: Input Table: _SPI_AG; Field Name: ‚KT'; Field Type: ‚TEXT
- **Calculate Field**: Input Table: SPI_AG; Field Name: ‚Shape Area'; Expression: Area
- **Calculate Field**: Input Table: SPI_AG; Field Name: ‚KT'; Expression
- **Expression**: Expression = 'AG'

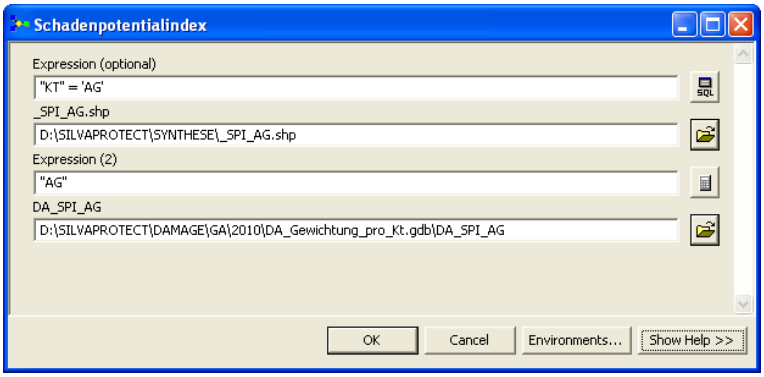


Abbildung 71: Nötige Informationen für die Berechnung des SPI.

Mit diesem Modell ist es möglich, die Fläche jedes einzelnen Schadenpotenzials, welches durch die oben beschriebenen Gefahrenprozesse betroffen ist, zu berechnen.

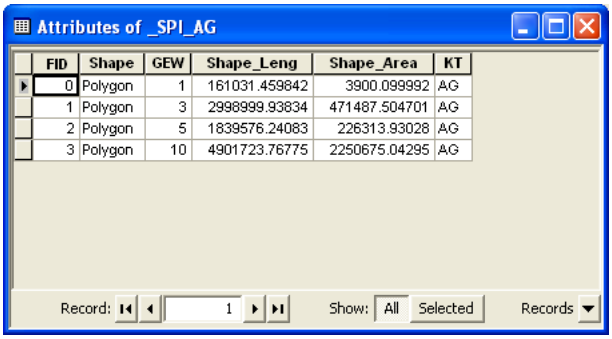


Abbildung 72: Attribute des relevanten Schadenpotenzials eines Kantons.

Diese Berechnung wurde danach für alle Kantone ausgeführt. Schliesslich wurden sämtliche Shapefiles zusammengefasst (GIS-Tool: MERGE) in einem Shapefile (SPI_CH), und die Attribut-Tabelle in eine EXCEL-Datei exportiert. Auf diese Weise konnte der SPI mit Hilfe einer Pivot-Tabelle direkt berechnet werden.

8 Literatur

(1) *ESRI, 2008*: ArcGIS Model Builder. Eine deutschsprachige Einführung zu Aufbau und Umgang mit Geoberarbeitungsmodellen in ArcGIS (d), 37 p. http://www.esri-germany.de/downloads/papers/ModelBuilder-Leitfaden.pdf

(2) Allen, D.W. 2011: Getting to Know ArcGIS ModelBuilder, 336 pages, Esri Press. http://store.esri.com/esri/showdetl.cfm?SID=2&Product_ID=1285&Category_ID=42

(3) Geo7, 2009: SilvaProtect-CH, Indentifizierung der Lawinenanrissgebiete, 4 p, unveröffentlicht.

(4) Geo7, 2006: SilvaProtect-CH, Datenmodell, 16 p., unveröffentlicht.

(5) Giamboni, M. 2008: SilvaProtect-CH Phase I, Projektdokumentation, 242 p.[1]

(6) Swisstopo, 2007: VECTOR25 Das digitale Landschaftsmodell der Schweiz, 31p. http://www.swisstopo.admin.ch/internet/swisstopo/de/home/products/landscape/vector25.html

(7) Swisstopo, 2012: Datenmodell Geologie, 151 p., http://www.geologieportal.ch/internet/geologieportal/de/home/knowledge/lookup/datamodel.html

[1] Diese Publikation kann bestellt werden unter der folgenden Adresse : gefahrenpraevention@bafu.admin.ch.

9 Anhänge

9.1 Datenmodell

Dieses Kapitel gibt eine Übersicht über das Datenmodell des Projekts SilvaProtect-CH und sein Inhalt ist im Detail beschrieben. Das Kapitel teilt sich in zwei Teile: Der erste Teil gibt eine Übersicht der verwendeten Daten sowie der berechnten Resultate. Der zweite Teil beschreibt den Dateninhalt für alle benötigten Modelle in den Kapiteln 4 bis 7.

9.1.1 Input und Output Daten

Die Abbildung 73 gibt eine Übersicht über die nötigen Daten und die Resultate. Jeder einzelne Datensatz ist im Detail beschrieben und enthält die folgenden Informationen:

- **Name**: Name
- **Properties**: Eigenschaften
- **Field Name**: Name des verwendeten Feldes
- **Value**: gesuchter oder verwendeter Wert im Feld
- **Data Type**: Datentyp

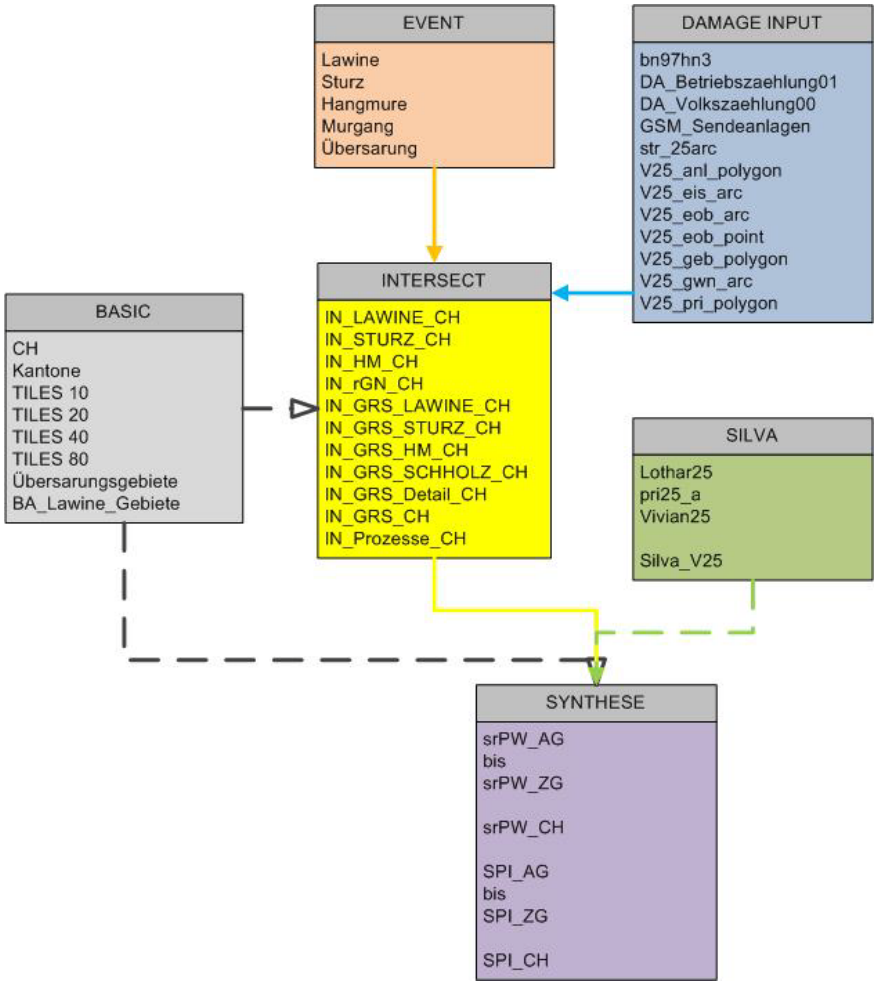


Abbildung 73: Datenmodell für SilvaProtect-CH.

Module BASIC

Name	Properties	Field Name	Value	Data Type
CH	Feature Class	Shape		Polygon
Kantone	Feature Class	Shape		Polygon
		KT	AG bis ZG	Text
TILES 10	Feature Class	Shape		Polygon
		TILES	1bc, 1bd bis 49cd	Text
TILES 20	Feature Class	Shape		Polygon
		TILES	1b, 1d bis 49c	Text
TILES 40	Feature Class	Shape		Polygon
		TILES_NR	1 bis 49	Text
		DA_GA	DA_GA_01 bis DA_GA_49	Text
TILES 80	Feature Class	Shape		Polygon
		TILES_NR	1 bis 11	Text
Uebersarungsgebiete	Feature Class	Shape		Polygon
		Name	Aare_Alpin, Inn, Jura, Mittelland_Nord, Mittelland_Sud, Reuss_Linth_Alpin, Rhein_Mitte, Rhein_Alpin, Rhone_Mittelland, Ticino	Text
BA_Lawine_Gebiete	Feature Class	Shape		Polygon
		Name	adu, bad, bas, ber, bie, bui, cha, com, dia, eig, fin, kes, lad, leo, lin, mat, muv, obe, pal, pil, rig, rot, sae, ter, tit, toe, tur, umb, wie, wil.	Text

Module EVENT (Lawine)

Name	Properties	Feature Class / Item Name	Value	Typology
xxxn_s_ph_r	Coverage	arc		Preliminary
xxxn_s_rel_r		Label		Not Applicable
xxxn_m_ph_r		polygon		Exists
xxxn_m_rel_r		tic		Not Applicable
xxxn_l_ph_r		region.ph		Exists
xxxn_l_rel_r				
xxxn_l_cap / point	Coverage	Shape		Geometry
		Area		Float
		Perimeter		Float
		xxxn_l_cap#		Binary
		xxxn_l_cap-ID		Binary
		NameID		Char
EV_Sturz_01 bis 49	Feature Class	Shape		Polyline
		Shape_Length		Double
EV_Hangmure_01 bis 49	Feature Class	Shape		Polyline
		Shape_Length		Double
EV_Murgang_01 bis 49	Feature Class	Shape		Polyline
		Shape_Length		Double
EV_Uebersarung_01 bis 49	Feature Class	Shape		Polyline
		Shape_Length		Double

Module DAMAGE (Input)

Name	Properties	Field Name	Value	Data Type
bn97hn3	Shapefile	Shape		Polygon
		GRIDCODE	15	Double
DA_Betriebszaehlung01	Shapefile	Shape		Point
		METER_X		Double
		METER_Y		Double
DA_Volkszaelung00	Shapefile	X_KOORD		Double
		Y_KOORD		Double
		A00WDTOT	0	Double
		A00WTZTOT	0	Double
GSM_Sendeanlagen	Shapefile	Shape		Point
		X		Double
		Y		Double
str_25arc	Shapefile	Shape		Polyline
		Length		Double
		OBJECTVAL	Autobahn, Autob_Ri, Autostr, Ein_Ausf, A_Zufahrt	Text
		TUNNELTYPE	„ - “	Text
V25_anl_polygon	Shapefile	Shape		Polygon
		Area		Doubel
V25_eis_arc	Shapefile	Shape		Polyline
		Length		Double
		OBJECTVAL	I_Geleis, Gt_Bahn, NS_BAHN1, SS_BAHN1, NS_BAHN2, SS_BAHN2	Text
		TUNNELTYPE	„ - “	Text
V25_eob_arc	Shapefile	Shape		Polyline
		Length		Double
		OBJECTVAL		Text
V25_eob_point	Shapefile	Shape		Point
		OBJECTVAL	Kamin, Reserv, W_Turm, Antenne, SendeAnl, Hafen, Schiffst, ARA, ElWerk	Text
V25_geb_polygon	Shapefile	Shape		Polygon
		Area	1963.5	Double
		OBJECTVAL	Z_Kuehlturm, Z_Lagertank, Z_WBecken, Z_Gasthof, Z_Gebaeude, Z_Huette Z_Schiessstand, Z_Treibhaus, Z_Innenhof, Z_Kirche, Z_Perron, Z_Schloss, Z_Station	Text
V25_eob_arc	Shapefile	Shape		Polyline
		Length		Double
		OBJECTVAL	Sender	Text
V25_gwn_arc	Shapefile	Shape		Polyline
		OBJECTVAL	Druckl_1, Druckl_2	Text
V25_pri_polygon	Shapefile	Shape		Polygon
		Shape_Area		Double
		OBJECTVAL	Z_Siedl	Text

Module INTERSECT

Name	Properties	Field Name	Value	Data Type
IN_LAWINE_CH	Shapefile	Shape		Polygon
		LAW	0, 1	Short Integer
IN_STURZ_CH	Shapefile	Shape		Polygon
		STURZ	0, 1	Short Integer
IN_HM_CH	Shapefile	Shape		Polygon
		HM	0, 1	Short Integer
IN_rGN_CH	Feature Class	Shape		Polygon
		Shape_Area		Double
IN_GRS_LAWINE_CH	Feature Class	Shape		Polygon
		GRS_LAW	0, 1	Double
IN_GRS_STURZ_CH	Feature Class	Shape		Polygon
		GRS_Sturz	0, 1	Double
IN_GRS_HM_CH	Feature Class	Shape		Polygon
		GRS_HM	0, 1	Double
IN_GRS_SCHHOLZ_CH	Feature Class	Shape		Polygon
		GRS_SchH	0, 1	Double
IN_GRS_Detail_CH	Shapefile	Shape		Polygon
		GRS_Sturz	0, 1	Short Integer
		GRS_SchH	0, 1	Short Integer
		GRS_LAW	0, 1	Short Integer
		GRS_HM	0, 1	Short Integer
IN_GRS_CH	Shapefile	Shape		Polygon
		GRS	0, 1	Short Integer
IN_Prozesse_CH	Shapefile	Shape		Polygon
		GRS	0, 1	Short Integer
		LAW	0, 1	Short Integer
		STURZ	0, 1	Short Integer
		HM	0, 1	Short Integer

Module SILVA

Name	Properties	Field Name	Value	Data Type
Lothar25	Feature Class	Shape		Polygon
		Shape_Area		Double
pri25_a	Feature Class	Shape		Polygone
		Shape_Area		Double
		OBJECTVAL	Z_GerWA, Z_GerWaO, Z_SumWa, Z_SumWaO, Z_Wald, Z_WaldOf	Text
Vivian25	Feature Class	Shape		Polygon
		Shape_Area		Double
pri25_a	Feature Class	Shape		Polygon
		Shape_Area		Double

Module SYNTHESE

Name	Properties	Field Name	Value	Data Type
srPW_XX	Shapefile	Shape		Polygon
		GRS	0, 1	Short Integer
		LAW	0, 1	Short Integer
		STURZ	0, 1	Short Integer
		HM	0, 1	Short Integer
		Shape_Area		Double
		KT	AG bis ZH	Text
SPI_XX	Shapefile	Shape		Polygon
		Gewichtung	1, 3, 5, 10	Short Integer
		Area		Double
		KT	AG bis ZH	Text

9.1.2 Beschreibung der einzelnen Modelle

9.1.2.1 Modelle für das Schadenpotenzial

Modell: Eisenbahn GA (Toolbox: SiPro 93 DAMAGE) (p. 25)

Input: V25_eis_arc.

Output: DA_Eisenbahn_GA_G1, DA_Eisenbahn_GA_G3, DA_Eisenbahn_GA_G5 und DA_Eisenbahn_swi.

Name	Properties	Field Name	Value	Data Type
DA_Eisenbahn_GA_G1	Shapefile	Shape		Polygon
		Gewichtung	1	Short Integer
		DA_Type	Eis	Text
DA_Eisenbahn_GA_G3	Shapefile	Shape		Polygon
		Gewichtung	3	Short Integer
		DA_Type	Eis	Text
DA_Eisenbahn_GA_G5	Shapefile	Shape		Polygon
		Gewichtung	5	Short Integer
		DA_Type	Eis	Text
DA_Eisenbahn_swi	Shapefile	Shape		Polygon
		DA_Eis	1	Short Integer

Modell: Anlagen GA (Toolbox: SiPro 93 DAMAGE) (p. 27)

Input: V25_gwn_arc, V25_eob_point, V25_eob_arc, GSM_Sendeanlagen, V25_pri_polygon, V25_geb_polygon und V25_anl_polygon

Output: DA_Anlage_GA_G1, DA_Anlage_GA_G3, DA_Anlage_GA_G10 und DA_Anlage_swi.

Name	Properties	Field Name	Value	Data Type
DA_Anlage_GA_G1	Shapefile	Shape		Polygon
		Gewichtung	1	Short Integer
		DA_Type	Anl	Text
DA_Anlage_GA_G3	Shapefile	Shape		Polygon
		Gewichtung	3	Short Integer
		DA_Type	Anl	Text
DA_Anlage_GA_G10	Shapefile	Shape		Polygon
		Gewichtung	10	Short Integer
		DA_Type	Anl	Text
DA_Anlage_swi	Shapefile	Shape		Polygon
		DA_Anl	1	Short Integer

Modell: Gebäude GA10 (Toolbox: SiPro 93 DAMAGE) (p. 33)

Input: DA_Volkszaehlung00

Output: DA_Gebauden_GA_G10 und DA_Gebauden_bew_swi.

Name	Properties	Field Name	Value	Data Type
DA_Gebauden_GA_G10	Shapefile	Shape		Polygon
		Gewichtung	10	Short Integer
		DA_Type	Geb	Text
DA_Gebauden_bew_swi	Shapefile	Shape		Polygon
		DA_Geb_bew	1	Short Integer

Modell: Gebäude GA5 temp (Toolbox: SiPro 93 DAMAGE) (p. 34)

Input: DA_Volkszaehlung00 und V25_pri_polygon

Output: DA_Gebauden_GA_G5temp und DA_Gebauden_temp_swi.

Name	Properties	Field Name	Value	Data Type
DA_Gebauden_GA_G5temp	Shapefile	Shape		Polygon
DA_Gebauden_temp_swi	Shapefile	Shape		Polygon
		DA_Geb_tmp	1	Short Integer

Modell: Gebäude GA5 ind punkt (Toolbox: SiPro 93 DAMAGE) (p. 35)

Input: DA_Betriebezaehlung und V25_geb_polygon

Output: DA_Gebauden_GA_G5ind_pkt.

Name	Properties	Field Name	Value	Data Type
DA_Gebauden_GA_G5ind_pkt	Shapefile	Shape		Polygon

Modell: Gebäude GA5 ind klein (Toolbox: SiPro 93 DAMAGE) (p. 36)

Input: DA_Betriebezaehlung und V25_geb_polygon

Output: DA_Gebauden_GA_G5ind_klein.

Name	Properties	Field Name	Value	Data Type
DA_Gebauden_GA_G5ind_klein	Shapefile	Shape		Polygon

Modell: **Gebäude GA5 ind gross (Toolbox: SiPro 93 DAMAGE) (p. 39)**

Input: DA_Betriebezaehlung und V25_geb_polygon

Output: DA_Gebauden_GA_G5ind_gross.

Name	Properties	Field Name	Value	Data Type
DA_Gebauden_GA_G5ind_gross	Shapefile	Shape		Polygon

Modell: **Gebäude GA5 ind areal (Toolbox: SiPro 93 DAMAGE) (p. 41)**

Input: bn97hn3, DA_Betriebezaehlung, DA_Volkzaehlung00 und V25_geb_polygon

Output: DA_Gebauden_GA_G5ind_areal.

Name	Properties	Field Name	Value	Data Type
DA_Gebauden_GA_G5ind_areal	Shapefile	Shape		Polygon

Modell: **Gebäude GA5 ind (Toolbox: SiPro 93 DAMAGE) (p. 43)**

Input: DA_gebauden_GA5 ind pkt, DA_gebauden_GA5 ind klein, DA_gebauden_GA5 ind_gross et DA_gebauden_GA5 ind areal

Output: DA_Gebauden_GA_G5ind et DA_Gebauden_ind_swi.

Name	Properties	Field Name	Value	Data Type
DA_Gebauden_GA_G5ind	Shapefile	Shape		Polygon
DA_Gebauden_ind_swi	Shapefile	Shape		Polygon
		DA_Ind	1	Short Integer

Modell: **Gebäude GA3 oeff (Toolbox: SiPro 93 DAMAGE) (p. 44)**

Input: V25_geb_polygon

Output: DA_Gebauden_GA_G3oeff und DA_Gebauden_oeff_swi.

Name	Properties	Field Name	Value	Data Type
DA_Gebauden_GA_G5oeff	Shapefile	Shape		Polygon
DA_Gebauden_oeff_swi	Shapefile	Shape		Polygon
		DA_Ind	1	Short Integer

Modell: Gebäude GA5 GA3 (Toolbox: SiPro 93 DAMAGE) (p. 45)

Input: DA_Gebauden_GA_G5ind, DA_Gebauden_GA_G5temp, DA_Gebauden_GA_G3_oeff und DA_Gebauden_GA_G10

Output: DA_Gebauden_GA_G5 und DA_Gebauden_GA_G3.

Name	Properties	Field Name	Value	Data Type
DA_Gebauden_GA_G5	Shapefile	Shape		Polygon
		Gewichtung	5	Short Integer
		DA_Type	Geb	Text
DA_Gebauden_GA_G3	Shapefile	Shape		Polygon
		Gewichtung	3	Short Integer
		DA_Type	Geb	Text

Modell: Strassen GA (Toolbox: SiPro 93 DAMAGE) (p. 46)

Input: Str_25_arc

Output: DA_Strasse_GA_G5, DA_Strasse_GA_G3, DA_Strassen_swi und DA_Autobahn_swi.

Name	Properties	Field Name	Value	Data Type
DA_Strasse_GA_G5	Shapefile	Shape		Polygon
		Gewichtung	5	Short Integer
		DA_Type	Str	Text
DA_Strasse_GA_G3	Shapefile	Shape		Polygon
		Gewichtung	3	Short Integer
		DA_Type	Str	Text
DA_Strasse_swi	Shapefile	Shape		Polygon
		DA_Str	1	Short Integer
DA_Autobahn_swi	Shapefile	Shape		Polygon
		DA_Autob	1	Short Integer

Modell: DA_GA (Toolbox: SiPro 93 DAMAGE) (p. 48)

Input: DA_Eisenbahn_swi, DA_Anlage_swi, DA_Gebauden_bew_swi, DA_Gebauden_temp_swi, DA_Gebauden_ind_swi, DA_Gebauden_oeff_swi, DA_Autobahn_swi und DA_Strassen_swi.

Output: DA_GA_att und DA_GA.

Name	Properties	Field Name	Value	Data Type
DA_GA_att	Shapefile	Shape		Polygon
		DA_Eis	0, 1	Short Integer
		DA_Geb_bew	0, 1	Short Integer
		DA_Geb_tmp	0, 1	Short Integer
		DA_Ind	0, 1	Short Integer
		DA_Geb_oef	0, 1	Short Integer
		DA_Autob	0, 1	Short Integer
		DA_Str	0, 1	Short Integer
		DA_Anl	0, 1	Short Integer
DA_GA	Shapefile	Shape		Polygone

Modell: DA_Gew_Kt_T1 **(Toolbox: SiPro 93 DAMAGE) (p. 50)**

Input: DA_Eisenbahn_GA_G1, DA_Eisenbahn_GA_G3, DA_Eisenbahn_GA_G5, DA_Anlage_GA_G1, DA_Anlage_GA_G3, DA_Anlage_GA_G10, DA_Gebauden_GA_G3, DA_Gebauden_GA_G5, DA_Gebauden_GA_G10, DA_Strassen_GA_G3 und DA_Strassen_GA_G5.

Modell: DA_Gew_Kt_T2 **(Toolbox: SiPro 93 DAMAGE) (p. 50)**

Input: DA_GA_10_Diss, DA_GA_5_Diss. DA_GA_3_Diss und DA_GA_1_Diss.

Modell: DA_Gew_Kt **(Toolbox: SiPro 93 DAMAGE) (p. 50)**

Input: DA_Gew_Kt_T1 und DA_Gew_Kt_T2.

Output: DA_SPI_AG.

Name	Properties	Field Name	Value	Data Type
DA_SPI_AG	Shapefile	Shape		Polygon
		Gewichtung	1, 3, 5, 10	Short Integer

9.1.2.2 Modelle für die relevanten Prozesse

Modell: DA_Lawine_pro_Gebiet **(Toolbox: SiPro 93 IN Lawine) (p. 57)**

Input: BA_Lawine_Gebiete und DA_GA.

Output: DA_adu, DA_bad … bis DA_wil.

Name	Properties	Field Name	Value	Data Type
DA_adu	Shapefile	Shape		Polygon

Script: pp_Setting (p. 59)

Input: adun_l_ph_r, adun_l_rel_r, adun_m_ph_r, adun_m_rel_r, adun_s_ph_r, adun_s_rel_r, adun_l_cap_point, DHM_10 und DA_adu.

Output: adun_rel_medium, adun_rel_small, adun_sw_large. Das Schlussresultat ist IN_LAWINE_CH.

Name	Properties	Field Name	Value	Data Type
adun_rel_medium	Shapefile	Shape		Polygon
		NAMEID	adun_mxx_rel	Text
adun_rel_small	Shapefile	Shape		Polygon
		NAMEID	adun_sxx_rel	Text
adun_sw_large	Shapefile	Shape		Polygon
		NAMEID	adun_lxx_rel	Text
IN_LAWINE_CH	Feature Class	Shape		Polygon
		LAW	1	Short Integer

Modell: IN Sturz07 (Toolbox: SiPro 93 IN STURZ) (p. 61)

Input: EV_Sturz_01 bis EV_Sturz_49, TILES10 und DA_GA_01 bis DA_GA_49.

Output: 7bd, 7da, 7db und 7dd.

Name	Properties	Field Name	Value	Data Type
5cc bis 5dd	Shapefile	Shape		Polyline

Modell: IN STURZ (Toolbox: SiPro 93 IN STURZ) (p. 63)

Input: IN Sturz01, IN Sturz02, … und IN Sturz49
EV_STURZ, TILES10 und DA_GA_01 bis DA_GA_49.

Output: 1bc, 1bd, bis 49cc und 49cd. Das Schlussresultat ist IN_STURZ_CH.

Name	Properties	Field Name	Value	Data Type
1bc bis 49cd	Shapefile	Shape		Polyline
IN_STURZ_CH	Feature Class	Shape		Polygon
		STURZ	1	Short Integer

Modell: IN HM01 (Toolbox: SiPro 93 IN HM) (p. 65)

Input: EV_HANGMURE_01 bis 49, TILES10 und DA_GA_01 bis DA_GA_49.

Output: 1bc, 1bd, 1cc, 1cd, 1da, 1db, 1dc und 1dd.

Name	Properties	Field Name	Value	Data Type
1bc bis 1dd	Shapefile	Shape		Polyline

Modell: IN HM (Toolbox: SiPro 93 IN HM) (p. 67)

Input: **IN HM 01, IN HM 02**, … und **IN HM 49**
EV_HANGMURE_01 bis 49, TILES10 und DA_GA_01 bis DA_GA_49.

Output: 1bc, 1bd, bis 49cc und 49cd. Das Schlussresultat ist IN_HM_CH.

Name	Properties	Field Name	Value	Data Type
1bc bis 49cd	Shapefile	Shape		Polyline
IN_HM_CH	Feature Class	Shape		Polygon
		HM	1	Short Integer

Modell: IN_Ueb (Toolbox: SiPro 93 IN_Ueb) (p.70)

Input: IN HM 01, IN HM 02, … und IN HM 49
EV_Uebersarung_xxx, EV_Uebersarung_start_xxx und DA_GA.

Output: IN_Uebersarung_start_xxx.

Name	Properties	Field Name	Value	Data Type
IN_Uebersarung_start_xxx	Shapefile	Shape		Point
		ID		Long Integer
		X		Long Integer
		Y		Long Integer

Modell: IN Mur 25 (Toolbox: SiPro 93 IN MUR) (p. 76)

Input: EV_MURGANG_01 bis EV_MURGANG_49, TILES10 und DA_GA_01 bis DA_GA_45.

Output: 25aa, 25ab, 25ac, 25ad, 25ba bis 25dd.

Name	Properties	Field Name	Value	Data Type
25aa jusqu'à 25dd	Shapefile	Shape		Polyline

Modell: IN MUR (Toolbox: SiPro 93 Murgang) (p. 78)

Input: IN Mur 01, IN Mur 02, … und IN Mur 49
EV_MURGANG_01 bis EV_MURGANG_49, TILES10 und DA_GA_01 bis DA_GA_49.

Output: 1bc, 1bd, … bis 49cc und 49cd. Das Schlussresultat ist IN_MUR_CH.

Name	Properties	Field Name	Value	Data Type
1bc jusqu'à 49cd	Shapefile	Shape		Polyline
IN_MUR_CH	Feature Class	Shape		Polygon
		HM	1	Short Integer

Bestimmung der relevanten Gerinneprozesse (p. 80)

Output: IN_RUENSE_UEB_CH, IN_RUNSE_MUR_CH und IN_rGN_CH.

Name	Properties	Field Name	Value	Data Type
IN_RUENSE_UEB_CH	Shapefile	Shape		Polyline
IN_RUNSE_MUR_CH	Shapefile	Shape		Polyline
IN_rGN_CH	Shapefile	Shape		Polyline

Relevante Gerinneprozesse (p. 84)

Die Schlussresultate sind IN_GRS_LAWINE_CH, IN_GRS_STURZ_CH, IN_GRS_HM_CH, IN_GRS_SchHolz_CH. IN_GRS_Detail_CH und IN_GRS_CH.

Name	Properties	Field Name	Value	Data Type
IN_GRS_Lawine_CH	Feature Class	Shape		Polygon
		GRS_Law	1	Short Integer
IN_GRS_Sturz_CH	Feature Class	Shape		Polygon
		GRS_Sturz	1	Short Integer
IN_GRS_HM_CH	Feature Class	Shape		Polygon
		GRS_HM	1	Short Integer
IN_GRS_SchHolz_CH	Feature Class	Shape		Polygon
		GRS_SchH	1	Short Integer
IN_GRS_Detail_CH	Shapefile	Shape		Polygon
		GRS_Sturz	0, 1	Short Integer
		GRS_SchH	0, 1	Short Integer
		GRS_Law	0, 1	Short Integer
		GRS_HM	0, 1	Short Integer
IN_GRS_CH	Shapefile	Shape		Polygon
		GRS	1	Short Integer

Zusammenfassung der relevanten Prozesse (p. 86)

Input: IN_Lawine_CH, IN_Sturz_CH, IN_HM_CH und IN_GRS_CH

Output: IN_Prozesse_CH.

Name	Properties	Field Name	Value	Data Type
IN_Prozesse_CH	Shapefile	Shape		Polygon
		GRS	0, 1	Short Integer
		LAW	0, 1	Short Integer
		STURZ	0, 1	Short Integer
		HM	0, 1	Short Integer

9.1.2.3 Modell für die Waldfläche

Modell: Silva (Toolbox: SiPro 93 Silva) (p. 87)

Input: pri25_a, lothar25 und vivian25.

Output: Silva_V25.

Name	Properties	Field Name	Value	Data Type
Silva_V25	Feature Class	Shape		Polygon

9.1.2.4 Modelle für die Synthese

Modell: Schutzwaldindex (Toolbox: SiPro 93 SYNTHESE) (p. 89)

Input: Kantone, Silva_V25_2010 und IN_Prozesse_2010.

Output: srPW_AG bis srPW_ZH und srPW_CH.

Name	Properties	Field Name	Value	Data Type
srPW_AG bis srPW_ZH	Shapefile	Shape		Polygon
		GRS	0, 1	Short Integer
		LAW	0, 1	Short Integer
		STURZ	0, 1	Short Integer
		HM	0, 1	Short Integer
		KT	AG bis ZH	Text
srPW_CH	Shapefile	Shape		Polygon
		GRS	0, 1	Short Integer
		LAW	0, 1	Short Integer
		STURZ	0, 1	Short Integer
		HM	0, 1	Short Integer
		KT	AG bis ZH	Text

Modell: Schadenpotenzialindex (Toolbox: SiPro 93 SYNTHESE) (p. 92)

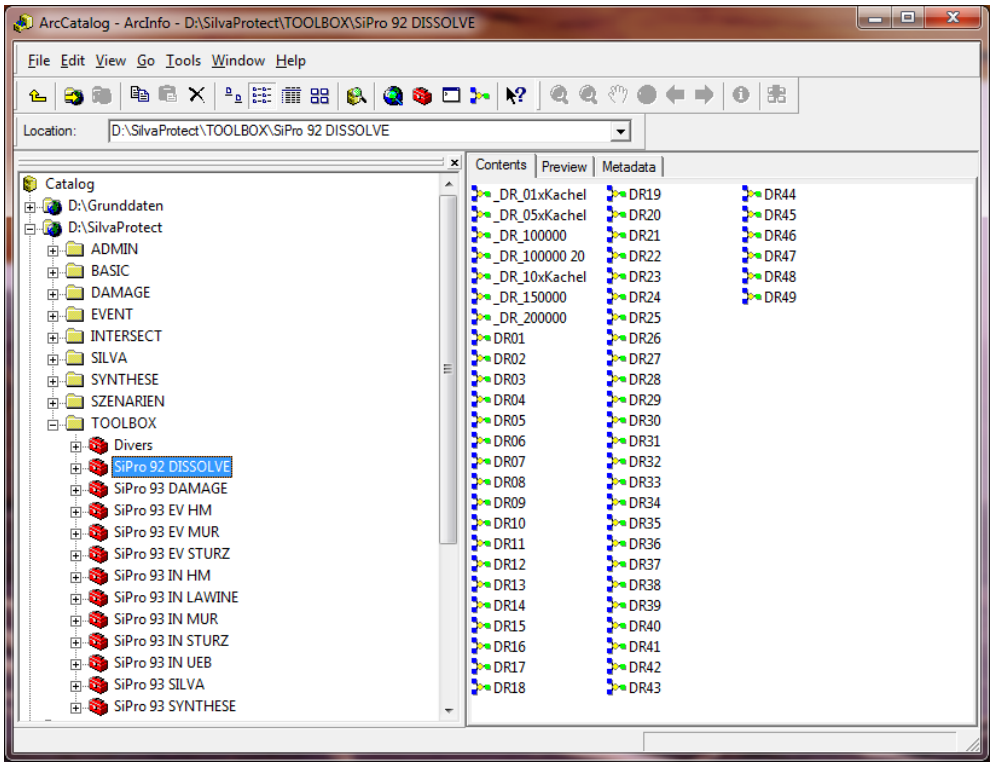
Input: Kantone, IN_Lawine_CH, IN_Murgang_CH, IN_Sturz_CH und IN_Uebersarung_CH.

Output: SPI_AG und SPI_CH.

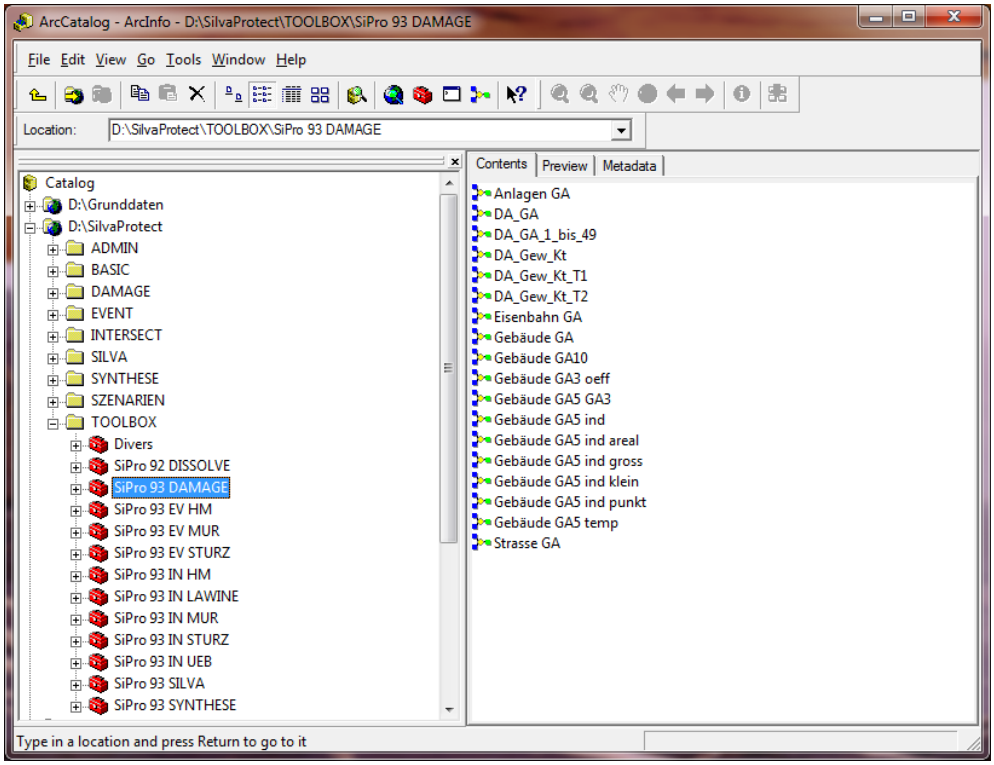
Name	Properties	Field Name	Value	Data Type
SPI_AG bis SPI_ZH	Shapefile	Shape		Polygon
		Gewichtung	1, 3, 5 ,10	Short Integer
		KT	AG bis ZH	Text
SPI_CH	Shapefile	Shape		Polygon
		Gewichtung	1, 3, 5 ,10	Short Integer
		KT	AG bis ZH	Text

9.2 Übersicht über die Toolbox des ModelBuilders

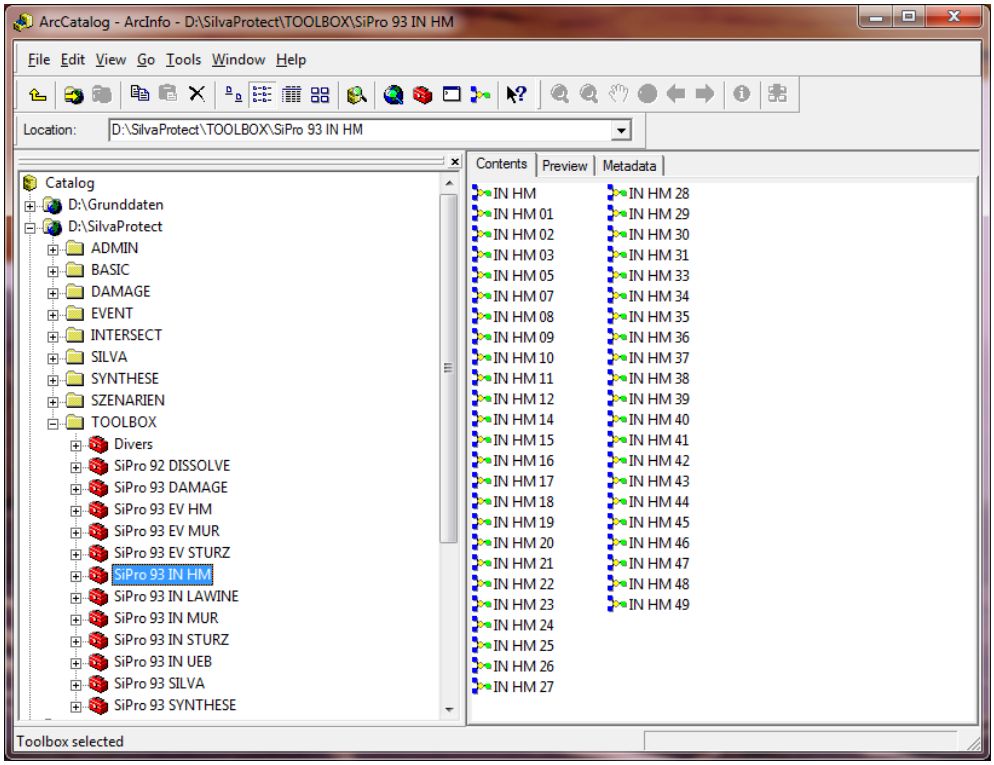
SiPro 92 Dissolve



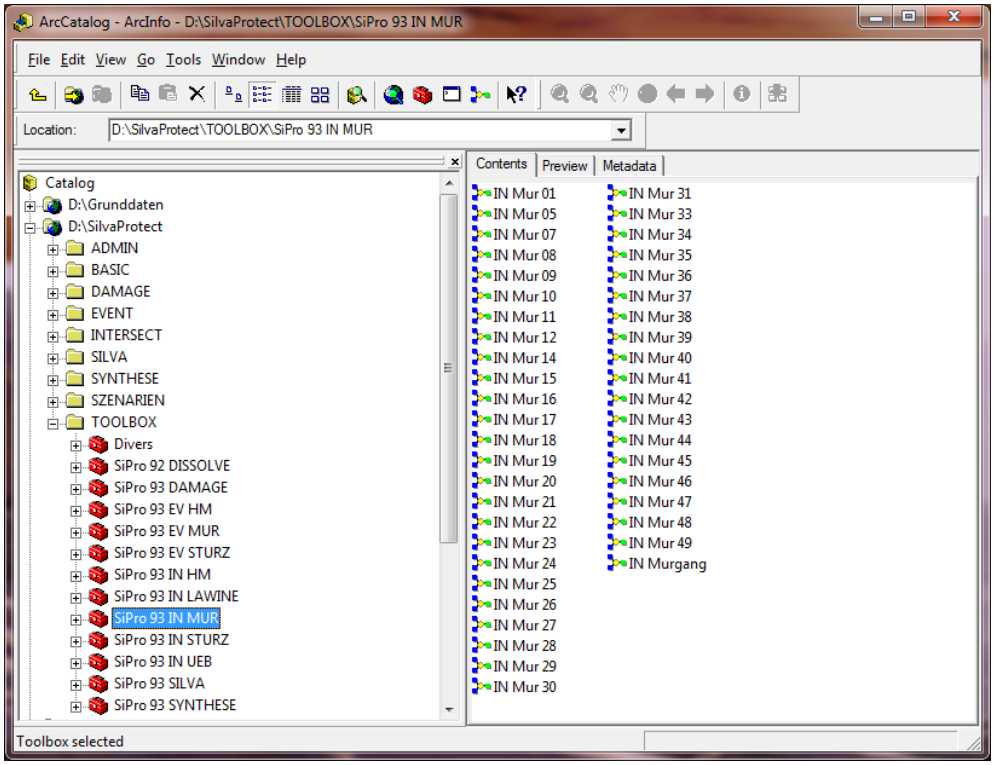
SiPro 93 DAMAGE



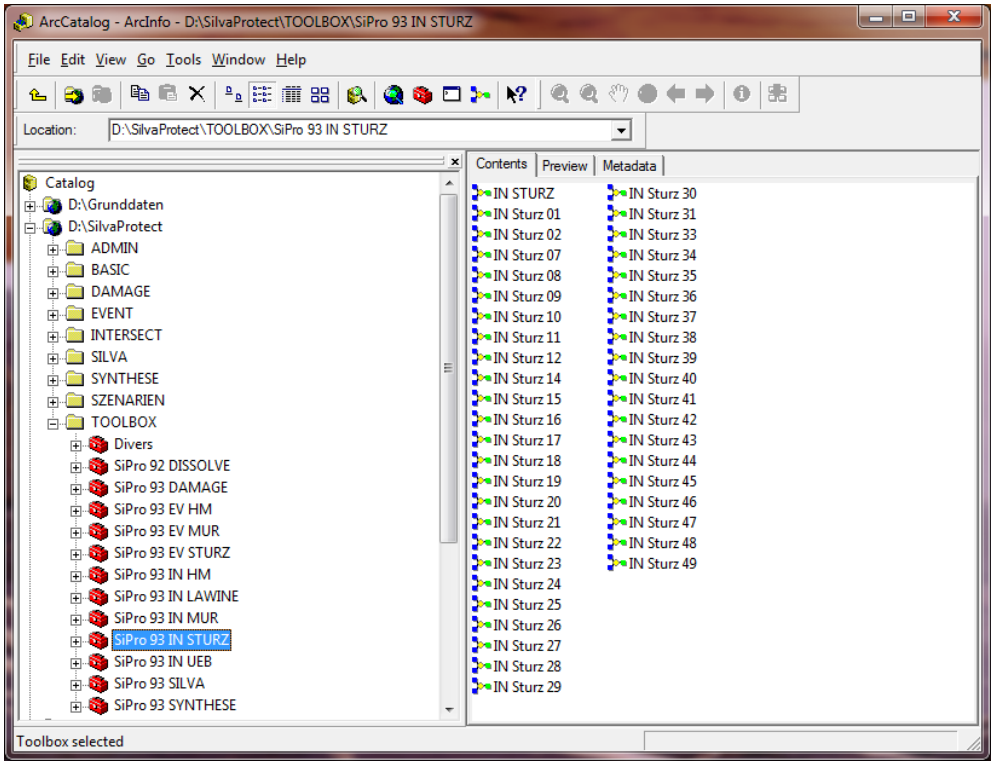
SiPro 93 IN HM



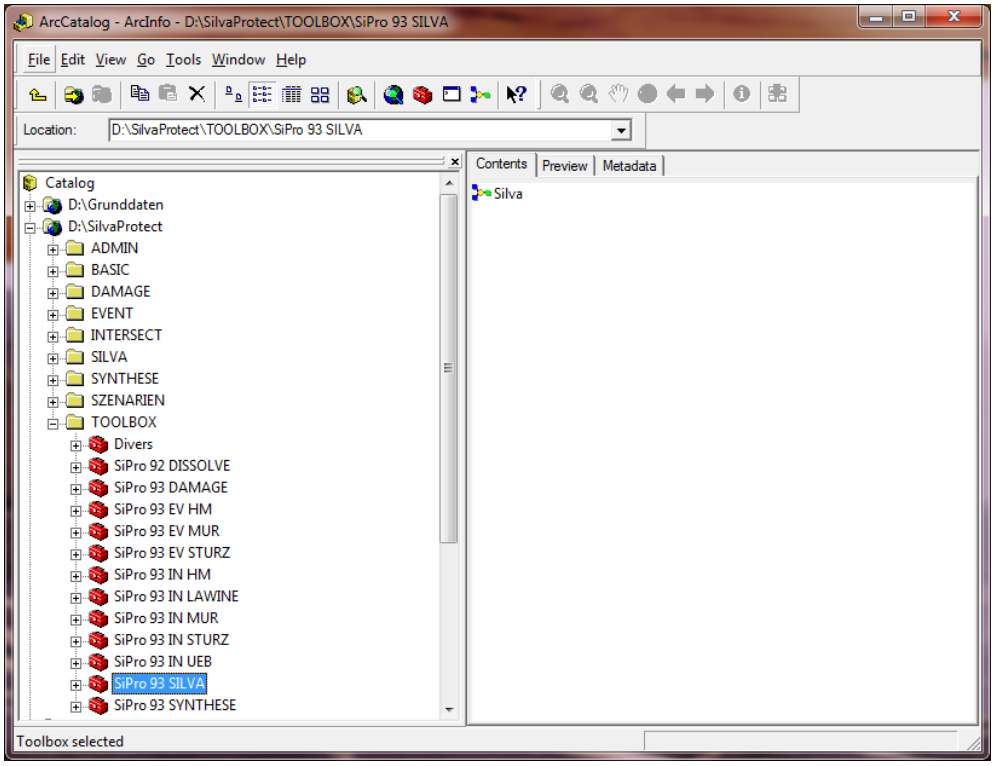
SiPro 93 IN MUR



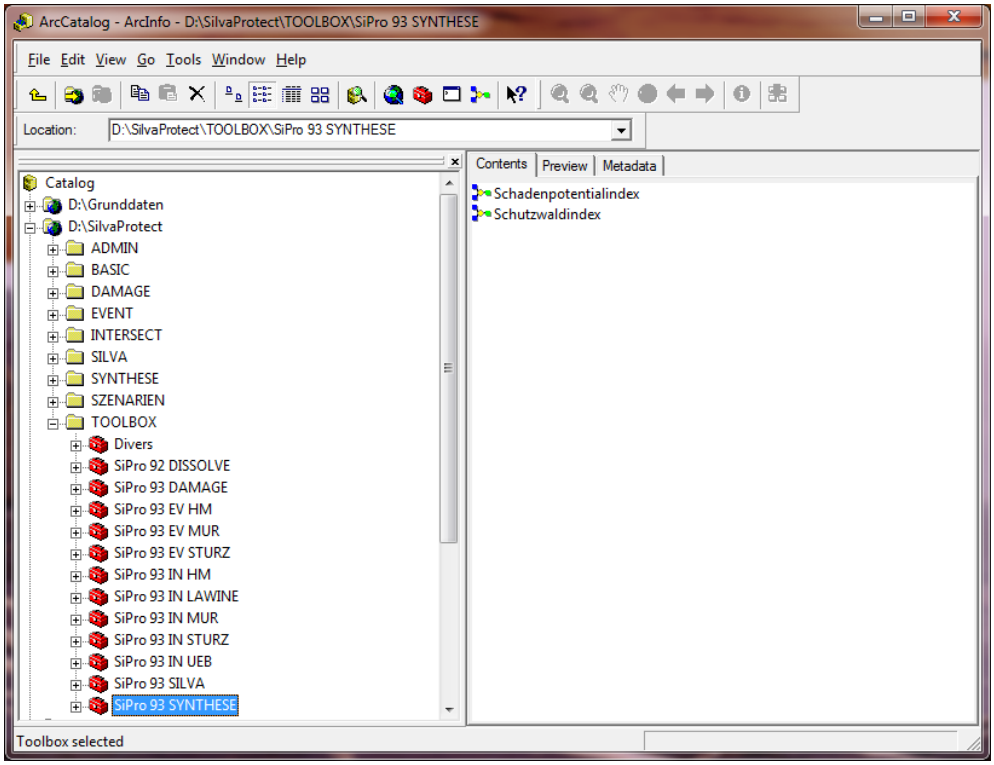
SiPro 93 IN STURZ



SiPro 93 SILVA



SiPro 93 SYNTHESE



9.3 Python Skript für die Lawine

9.3.1 Pp_Launch.py: Start

```
#-*- coding: utf-8 -*-
# pp_Launch.py
'''
@summary: Identifikation schadenpotenzialsrelevanter Lawinenanrissgebiete Silvaprotect.

Description:
Identifikation der schadenpotenzialsrelevanten Lawinenperimeter(Silvaprotect PH)
und der dazugehÃ¶rigen Anrissgebiete (Silvaprotect REL)
fÃ¼r die Kategorien small, medium und large.

History:
Postprocessing_small_med_large_v2_04.py WSL
Migration nach ArcGIS 9.1 geo7
Migration nach ArcGIS 9.3.1 geo7

Family:
pp_Launch.py (aufrufendes Skript)
pp_Program.py (AuslÃ¶sung einzelner Verarbeitungsschritt)
pp_ proc_rel.py (alle Verarbeitungsschritte und Methoden)
pp_Settings.py (Parametrisierung und Einstellungen)
pp_util_GIS.py (Aufruf gp-Objekt, Benachrichtigungen)

Dokumentation:
Workflow und Produkte sind beschrieben in:
Identifikation schadenpotenzialsrelevanter Lawinenanrissgebiete Silvaprotect
geo7, 04.7.2011

@version: 20110704 2.0
@author: Peter Gsteiger (gsp), Andri Baltensweiler (WSL) Originalscript
@note: Der Ablauf ist in __main__ abgebildet.
@note: Wird das Skript ohne Parameter aufgerufen, werden die Parameter in pp_Settings.py gelesen.

@param1: src_Workspace (workspace), Ablage der Coverages mit PH und REL regions
@param2: tar_Folder (Folder), Zielverzeichnis (lokal) zur Ablage der Ergebnisse
@param3: gebiet, (String,3 Buchstaben), GebietskÃ¼rzel
@param4: DamLayer (Feature Class), Schadenpotenzial, Polygone)
@param5: dtm (Rasterdataset), DTM Silvaprotect

'''


# ----------   Import system modules
import sys
import os
import codecs
import gc
import time,datetime


# ----------   Import delivered modules
import pp_Settings
import pp_util_GIS

# ----------   Vars
_path_python = pp_Settings._path_python
_path_script = pp_Settings._path_script



# ---------------------------------------------------------------------------------------
def check_exit(succ_value):
    if succ_value == 1:
        scriptmsg = ">>> Prozess Abbruch " + str(time_tar_gdb) + " in " + Verarbeitung
        pp_util_GIS.gpmsg(gp,scriptmsg,0)
        sys.exit(succ_value)
    else:
        scriptmsg = Verarbeitung + " ok "
        pp_util_GIS.gpmsg(gp,scriptmsg,0)



# ----------   Main-Procedure
if __name__ == '__main__':
```

```
    '''
    '''

    # ----------------   Zeitstempel Verarbeitungsbeginn
    starttime =  time.clock()
    time_tar_gdb =  time.strftime("%Y%m%d%H%M%S")
    #time_tar_gdb =  str(20110704003209)


    # ----------------   Defaults
    gp = None
    Verarbeitung = ""

    # ----------------   Eingabe Parameter
    src_Workspace = "#"
    tar_Folder = "#"
    gebiet = "#"
    DamLayer = "#"
    dtm = "#"

    try:
        src_Workspace = sys.argv[1]
        tar_Folder = sys.argv[2]
        gebiet = sys.argv[3].lower()
        DamLayer = sys.argv[4]
        dtm = sys.argv[5]
    except:
        pass

    if src_Workspace == "#":
        src_Workspace = pp_Settings.src_Workspace
    if tar_Folder == "#":
        tar_Folder = pp_Settings.tar_Folder
    if gebiet == "#":
        gebiet = pp_Settings.gebiet.lower()
    if DamLayer == "#":
        DamLayer = pp_Settings.DamLayer
    if dtm == "#":
        dtm = pp_Settings.dtm


    # -------------------  Abarbeiten Prozesse
    param = [_path_python + "/python.exe", _path_script + "/pp_Program.py", src_Workspace, tar_Folder, gebiet,
DamLayer, dtm, str(time_tar_gdb)]
    succ_value = 0
    nameidfile = tar_Folder + "/" + gebiet + str(time_tar_gdb) + "/" + gebiet + ".txt"

    scriptmsg = ">>>> Prozess Start " + str(time_tar_gdb)
    pp_util_GIS.gpmsg(gp,scriptmsg,0)

    scriptmsg = "Parameter zu Verarbeitung " + str(time_tar_gdb) + ":"
    pp_util_GIS.gpmsg(gp,scriptmsg,0)
    scriptmsg = "src_Workspace: " + src_Workspace
    pp_util_GIS.gpmsg(gp,scriptmsg,0)
    scriptmsg = "tar_Folder: " + tar_Folder
    pp_util_GIS.gpmsg(gp,scriptmsg,0)
    scriptmsg = "gebiet: " + gebiet
    pp_util_GIS.gpmsg(gp,scriptmsg,0)
    scriptmsg = "DamLayer: " + DamLayer
    pp_util_GIS.gpmsg(gp,scriptmsg,0)
    scriptmsg = "dtm: " + dtm
    pp_util_GIS.gpmsg(gp,scriptmsg,0)


    # -------------------
    Verarbeitung = "CREATE_WORKSPACES"
    scriptmsg = ">> " + Verarbeitung
    pp_util_GIS.gpmsg(gp,scriptmsg,0)
    try:
        os.spawnv(os.P_WAIT, _path_python + "/python.exe", param + [Verarbeitung, ""])
    except Exception, e:
        succ_value = 1
    check_exit(succ_value)

    # -------------------
    Verarbeitung = "BASISDATEN"
    scriptmsg = ">> " + Verarbeitung
```

```
pp_util_GIS.gpmsg(gp,scriptmsg,0)
try:
   os.spawnv(os.P_WAIT, _path_python + "/python.exe", param + [Verarbeitung, ""])
except Exception, e:
   succ_value = 1
check_exit(succ_value)

# ------------------
Verarbeitung = "PROC_SPOTREL_PH"
scriptmsg = ">> " + Verarbeitung
pp_util_GIS.gpmsg(gp,scriptmsg,0)
try:
   os.spawnv(os.P_WAIT, _path_python + "/python.exe", param + [Verarbeitung, ""])
except Exception, e:
   succ_value = 1
check_exit(succ_value)

# ------------------
Verarbeitung = "RDS_BASIS_GEBIET"
scriptmsg = ">> " + Verarbeitung
pp_util_GIS.gpmsg(gp,scriptmsg,0)
try:
   os.spawnv(os.P_WAIT, _path_python + "/python.exe", param + [Verarbeitung, ""])
except Exception, e:
   succ_value = 1
check_exit(succ_value)

# ------------------
Verarbeitung = "GET_NAMEIDS"
scriptmsg = ">> " + Verarbeitung
pp_util_GIS.gpmsg(gp,scriptmsg,0)
try:
   os.spawnv(os.P_WAIT, _path_python + "/python.exe", param + [Verarbeitung])
except Exception, e:
   succ_value = 1
check_exit(succ_value)

# ------------------
Verarbeitung = "PROC_NAMEIDS"
scriptmsg = ">> " + Verarbeitung
pp_util_GIS.gpmsg(gp,scriptmsg,0)
try:
   file = open(nameidfile, "r")
   lines = file.readlines()
   file.close()

   anznames = len(lines)
   ncount = 0

   for line in lines:
      ncount = ncount + 1
      nameid = line.replace("\n", "")
      scriptmsg = "verarbeite NAMEID: " + nameid + " " + str(ncount) + " von " + str(anznames)
      pp_util_GIS.gpmsg(gp,scriptmsg,0)

      os.spawnv(os.P_WAIT, _path_python + "/python.exe", param + [Verarbeitung, nameid])
except Exception, e:
   succ_value = 1
check_exit(succ_value)


# ------------------
Verarbeitung = "PROC_SPOTREL_REL"
scriptmsg = ">> " + Verarbeitung
pp_util_GIS.gpmsg(gp,scriptmsg,0)
try:
   os.spawnv(os.P_WAIT, _path_python + "/python.exe", param + [Verarbeitung, ""])
except Exception, e:
   succ_value = 1
check_exit(succ_value)


if succ_value == 0:
   scriptmsg = ">>> Prozess Ende " + str(time_tar_gdb) + " ok"
   pp_util_GIS.gpmsg(gp,scriptmsg,0)

sys.exit(succ_value)
```

9.3.2 Pp_Programm.py: Auslöser für die verschiedenen Etappen

```
#-*- coding: utf-8 -*-
#-------------------------------------------------------------------------------
# pp_Program.py

'''
'''

# ----------------   Import system modules
import sys
import os
import time,datetime


# ----------------   Import delivered modules
import pp_Settings
import pp_util_GIS
import pp_proc


if __name__ == '__main__':


    # ------------------  Parameter

    try:
        src_Workspace = str(sys.argv[1])
    except:
        src_Workspace = pp_Settings.src_Workspace

    try:
        tar_Folder = sys.argv[2]
    except:
        tar_Folder = pp_Settings.tar_Folder

    try:
        gebiet = str(sys.argv[3])
    except:
        gebiet = pp_Settings.gebiet.lower()

    try:
        DamLayer = sys.argv[4]
    except:
        DamLayer = pp_Settings.DamLayer

    try:
        dtm = sys.argv[5]
    except:
        dtm = pp_Settings.dtm

    try:
        time_tar_gdb = sys.argv[6]
    except:
        time_tar_gdb =  time.strftime("%Y%m%d%H%M%S")

    try:
        Verarbeitung = str(sys.argv[7])
    except:
        Verarbeitung = "CREATE_WORKSPACES"
        Verarbeitung = "BASISDATEN"
        Verarbeitung = "PROC_SPOTREL_PH"
        Verarbeitung = "RDS_BASIS_GEBIET"
        Verarbeitung = "GET_NAMEIDS"
        Verarbeitung = "PROC_NAMEIDS"
        Verarbeitung = "PROC_SPOTREL_REL"
        Verarbeitung = "RDS_BASIS_GEBIET"
        Verarbeitung = "PROC_SPOTREL_PH"

    try:
        nameid = sys.argv[8]
    except:
        nameid =  "umbn_l10_ph"

    pp = pp_proc.pp_proc(src_Workspace, tar_Folder, gebiet, DamLayer, dtm, time_tar_gdb, Verarbeitung, na-
meid)
    pp.Prozesssteuerung()
```

9.3.3 Pp_proc_rel.py: Alle Berechnungsetappen und Methodik

```
#-*- coding: utf-8 -*-
'''
pp_proc.py


'''

# ----------------   Import system modules
import sys, os, time, string, copy


# ----------------   Import delivered modules
import pp_Settings
import pp_util_GIS


class pp_proc:

    # ---------------------------------------------------------------------------
    def __init__(self, __src_Workspace, __tar_Folder, __gebiet, __DamLayer, __dtm, __time_tar_gdb,
__Verarbeitung, __nameid):


        # ------------------  Globals
        self.src_Workspace = __src_Workspace
        self.tar_Folder = __tar_Folder
        self.gebiet = __gebiet
        self.DamLayer = __DamLayer
        self.dtm = __dtm
        self.time_tar_gdb = __time_tar_gdb
        self.Verarbeitung = __Verarbeitung
        self.nameid = __nameid

        self.gebiet = self.gebiet.lower()
        self.gebiet = self.gebiet.strip(" ")



        # ----------------   Vars
        self.ph_string = pp_Settings.ph_string.lower()
        self.rel_string = pp_Settings.rel_string.lower()
        self.cap_string = pp_Settings.cap_string.lower()
        self.wald_string = pp_Settings.wald_string.lower()

        self.tar_gdbname = self.gebiet + self.time_tar_gdb + ".gdb"
        self.tar_gdb = self.tar_Folder + "/" + self.tar_gdbname
        self.rds_folder = self.tar_Folder + "/" + self.gebiet + self.time_tar_gdb

        #self.tar_gdb = r"C:\Workspace\KPRO_LASCRIPT2011\umb20110704160856.gdb"
        #self.rds_folder = r"C:\Workspace\KPRO_LASCRIPT2011\umb20110704160856"

        self.srcFC_spot = self.tar_gdb + "/" + self.gebiet + "_spot"
        self.srcFC_cap = self.tar_gdb + "/" + self.gebiet + "_cap"
        self.srcFC_ph = self.tar_gdb + "/" + self.gebiet + self.wald_string + "_" + self.ph_string + "_" + "large"
        self.srcFC_rel = self.tar_gdb + "/" + self.gebiet + self.wald_string + "_" + self.rel_string + "_" + "large"

        self.nameidfile = self.rds_folder + "/" + self.gebiet + ".txt"

        self.Aspect = self.rds_folder + "/" + "aspect"
        self.aspRad = self.rds_folder + "/" + "asprad"
        self.aspCosRad = self.rds_folder + "/" + "aspcosrad"
        self.aspSinRad = self.rds_folder + "/" + "aspsinrad"
        self.Slope = self.rds_folder + "/" + "slope"
        self.mainSlpDir = self.rds_folder + "/" + "mainslpdir"
        self.SPgd = self.rds_folder + "/" + "spgd"


        # ------------------  self.gp-Aufruf und self.gp-Settings
        ARCLICREQ = pp_Settings.ARCLICREQ
        self.gp = None
        ARCGIS_DIR = os.path.expandvars("$ARCGISHOME")
        self.gp = pp_util_GIS.GIS_Setup(ARCLICREQ)

        # Verfuegbarkeit self.gp-Objekt
        if self.gp == None:
```

```
            scriptmsg = "Abbruch! Das Geoprocessor Objekt konnte nicht erstellt werden! "
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()
        # Pruefen der eingestellten Lizenzierung
        else:
            lic_dt = self.gp.ProductInfo()

            if string.lower(ARCLICREQ) <> string.lower(lic_dt):
                scriptmsg = "ArcGIS-Lizenz nicht verfuegbar: " + str(ARCLICREQ)
                pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
                sys.exit()
            else:
                scriptmsg = "ArcGIS-Lizenz: " + str(ARCLICREQ)
                pp_util_GIS.gpmsg(self.gp,scriptmsg,0)

        self.gp.OverWriteOutput = 1
        pp_util_GIS.gdbsettings(self.gp)
        return



    # ---------------------------------------------------------------------------
    def Prozesssteuerung(self):

        '''
        @summary: Steuerung  self.Verarbeitung
        @summary: unerwartete Abbrueche werden vom aufrufenden Skript NGKLU_Produkte_PyLaunch.py abgefangen
        '''

        # ----------------   Validierung Parameter
        # src_Workspace pruefen
        if not self.gp.exists(self.src_Workspace):
            scriptmsg = "Abbruch! Workspace " + self.src_Workspace + " existiert nicht."
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()


        # self.tar_Folder pruefen
        if not os.path.isdir(self.tar_Folder):
            scriptmsg = "Abbruch! Folder " + self.tar_Folder + " existiert nicht."
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()


        # self.DamLayer pruefen
        if not self.gp.exists(self.DamLayer):
            scriptmsg = "Abbruch! Schadenpotenzial " + self.DamLayer + " existiert nicht."
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()

        # self.dtm pruefen
        if not self.gp.exists(self.dtm):
            scriptmsg = "Abbruch! dtm " + self.dtm + " existiert nicht."
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()

        # ---------------------------------------------------------------------------
        if  self.Verarbeitung == "CREATE_WORKSPACES":
            self.CREATE_WORKSPACES()

        # ---------------------------------------------------------------------------
        if  self.Verarbeitung == "BASISDATEN":
            self.BASISDATEN()

        # ---------------------------------------------------------------------------
        if  self.Verarbeitung == "PROC_SPOTREL_PH":
            self.PROC_SPOTREL_PH()

        # ---------------------------------------------------------------------------
        if  self.Verarbeitung == "RDS_BASIS_GEBIET":
            self.RDS_BASIS_GEBIET()

        # ---------------------------------------------------------------------------
        if  self.Verarbeitung == "GET_NAMEIDS":
            self.GET_NAMEIDS()
```

```
        # ---------------------------------------------------------------------------
        if  self.Verarbeitung == "PROC_NAMEIDS":
            self.PROC_NAMEIDS()

        # ---------------------------------------------------------------------------
        if  self.Verarbeitung == "PROC_SPOTREL_REL":
            self.PROC_SPOTREL_REL()

        del self.gp
        return


    # ---------------------------------------------------------------------------
    def PROC_SPOTREL_REL(self):
        scriptmsg = "berechne schadenpotenzialsrelevante Anrissgebiete REL "
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)


        #find phSuffix in the first fc
        self.gp.workspace = self.tar_gdb
        searchstring = "*_ws"
        fcs = self.gp.ListFeatureClasses(searchstring,"POLYGON")
        wsfclist = []
        for entry in fcs:
            wsfclist.append(entry)



        # Die berechneten Einzugsgebiete werden mit den zugehoerigen rel Features verschnitten.
        # Liste der Einzugsgebiete erstellen

        count = 0
        swFClist = []
        appendexpr = ""

        fielddef_nameidws = ["nameidws","TEXT","","","","","NULLABLE","NON_REQUIRED",""]
        fielddef_relOK = ["relOK","SHORT","","","","","NULLABLE","REQUIRED",""]
        fieldlist = [fielddef_nameidws, fielddef_relOK]

        for wsFC in wsfclist:

            count = count + 1

            for mylist in fieldlist:
                if len(self.gp.ListFields(wsFC,mylist[0]))==0:
                    self.gp.AddField_management(wsFC,                                                                    myl-
ist[0],mylist[1],mylist[2],mylist[3],mylist[4],mylist[5],mylist[6],mylist[7],mylist[8])

            myfcname = os.path.basename(wsFC)
            nameidbase = string.replace(myfcname.lower(), "_ws", "")
            nameidsw = nameidbase + "_sw"
            self.gp.CalculateField_management(wsFC, "nameidws", """ + nameidsw + """, "PYTHON", "")

            fcname = self.gebiet + self.wald_string + "_seg_"  + "large"
            wsFCout = self.tar_gdb + "/" + fcname
            if count == 1:
                scriptmsg = "Transfer Einzugsgebiete large in FC: " + os.path.basename(wsFCout)
                pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
                self.cleanup(wsFCout)
                self.gp.CreateFeatureclass(self.tar_gdb, fcname, "POLYGON", wsFC)
                appendexpr = wsFC
            else:
                appendexpr = appendexpr + "; " + wsFC

        self.gp.Append_management(appendexpr, wsFCout, "NO_TEST")

        tmpFCrel = self.tar_gdb + "/x_intersect"
        self.cleanup(tmpFCrel)
        intersectstring = self.srcFC_rel + ";" + wsFCout
        self.gp.Intersect_analysis(intersectstring, tmpFCrel)



        # In einem update cursor werden die rel Flaechen mit korrespondierenden Namen
        # zu wsFC markiert
        scriptmsg = "markiere zugehoerige rel in " + os.path.basename(tmpFCrel)
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
```

```
        rows = self.gp.UpdateCursor(tmpFCrel)
        row = rows.Next()
        while row:
            # Eintraege nameid auslesen
            # in nameid von rel steht nameid zu rel Layer
            nid1 = row.GetValue("nameid")
            name1 = string.replace(nid1.lower(), "_" + self.rel_string.lower(), "")
            nid2 = row.GetValue("nameidws")
            name2 = string.replace(nid2.lower(), "_sw", "")
            #print name1 + " " + name2
            if name1 == name2:
                row.relOK = "1"
            rows.UpdateRow(row)
            row = rows.Next()
        del row, rows


        # Reduktion auf zulaessige rel Flaechen
        scriptmsg = "Reduktion auf zulaessige rel Flaechen "
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
        name = "x" + self.gebiet + self.wald_string + "_sw_"  + "large"
        #print name
        swFC = self.tar_gdb + "/" + name
        self.cleanup(swFC)
        selexpr = "relOK" + " = " + "1"
        self.gp.FeatureClassToFeatureClass_conversion(tmpFCrel, self.tar_gdb, name, selexpr)


        name = self.gebiet + self.wald_string + "_sw_"  + "large"
        outFCrel = self.tar_gdb + "/" + name
        self.cleanup(outFCrel)
        self.gp.Dissolve_management(swFC, outFCrel, "nameid", "", "SINGLE_PART")
        scriptmsg = "schreibe Ergebnis "  + os.path.basename(outFCrel)
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)

        self.cleanup(swFC)
        self.cleanup(tmpFCrel)
        self.cleanup(wsFCout)

        return

    # ---------------------------------------------------------------------------------------
    def PROC_NAMEIDS(self):

        nameid_base = string.replace(self.nameid.lower(), "_" + self.ph_string.lower(), "")
        capname = nameid_base + self.cap_string




        # ph Features zu nameid separieren
        anzpolys = 0
        fc_ph = self.tar_gdb + "/" + self.ph_string + "_" + self.nameid
        self.cleanup(fc_ph)
        scriptmsg = "berechne " + os.path.basename(fc_ph)
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
        selExpr1 = "NAMEID" + " = " + "'" + self.nameid + "'"
        self.gp.FeatureClassToFeatureClass_conversion(self.srcFC_ph,   self.tar_gdb,   os.path.basename(fc_ph),
selExpr1)
        obj_count = self.gp.GetCount_management(fc_ph)
        anzpolys = int(obj_count.GetOutput(0))
        if anzpolys == 0:
            scriptmsg = "Abbruch! PH nameid: " + self.nameid + ", kein ph Poly selektiert "
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()

        # cap Features zu nameid separieren
        anzpts = 0
        fc_cap = self.tar_gdb + "/" + self.cap_string + "_" + self.nameid
        self.cleanup(fc_cap)
        scriptmsg = "berechne " + os.path.basename(fc_cap)
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
        selExpr2 = "NAMEID" + " = " + "'" + capname + "'"
        self.gp.FeatureClassToFeatureClass_conversion(self.srcFC_cap, self.tar_gdb, os.path.basename(fc_cap),
selExpr2)
        obj_count = self.gp.GetCount_management(fc_cap)
        anzpts = int(obj_count.GetOutput(0))
```

```
        if anzpts == 0:
            scriptmsg = "Abbruch! PH nameid: " + self.nameid + ", keine cap Punkte mit NAMEID = " + capname + "
gefunden."
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()

        self.gp.extent = self.setextent(fc_ph)
        self.gp.snapraster = self.dtm



        # create TIN

        scriptmsg = "Tin " + nameid_base
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
        xoutTin = self.rds_folder + "/" + "xtin"
        self.cleanup(xoutTin)
        self.gp.createtin_3d(xoutTin)
        # edit TIN
        editstring = fc_cap + " " + "ANGLE" + " " + "<none> masspoints false"
        self.gp.EditTin_3d(xoutTin, editstring)



        scriptmsg = "Rasterverarbeitung " + nameid_base
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)

        zoneGD = self.rds_folder + "/" + "zgd"
        self.cleanup(zoneGD)
        #self.gp.FeatureToRaster_conversion(fc_ph,"nameid",zoneGD,"10")
        self.gp.FeatureToRaster_conversion(fc_ph,"objectid",zoneGD,"10")

        # mimimal benoetigten Extent festlegen

        self.gp.extent = self.setextent(zoneGD)
        self.gp.snapraster = self.dtm

        xTinRaster = self.rds_folder + "/" + nameid_base
        self.cleanup(xTinRaster)
        self.gp.TinRaster_3D(xoutTin, xTinRaster, "FLOAT", "LINEAR", "CELLSIZE 10", "1")

        '''

        rgrp = "rgrp"
        self.cleanup(rgrp)
        #self.gp.RegionGroup_sa(zoneGD, rgrp, "EIGHT", "WITHIN", "NOLINK")
        self.gp.RegionGroup_sa(zoneGD, rgrp, "EIGHT")

        '''

        AspectRange = self.rds_folder + "/" + "AspectRange"
        self.cleanup(AspectRange)
        #self.gp.ZonalStatistics_sa(fc_ph, "nameid", Aspect, AspectRange, "RANGE", "DATA")
        self.gp.ZonalStatistics_sa(zoneGD, "value", self.Aspect, AspectRange, "RANGE", "DATA")


        x = self.rds_folder + "/" + "zmAcosr"
        self.cleanup(x)
        #self.gp.ZonalStatistics_sa(fc_ph, "nameid", aspCosRad, x, "MEAN", "DATA")
        #self.gp.ZonalStatistics_sa(rgrp, "value", self.aspCosRad, x, "MEAN", "DATA")
        self.gp.ZonalStatistics_sa(zoneGD, "value", self.aspCosRad, x, "MEAN", "DATA")

        y = self.rds_folder + "/" + "zmAsinr"
        self.cleanup(y)
        #self.gp.ZonalStatistics_sa(fc_ph, "nameid", aspSinRad, y, "MEAN", "DATA")
        #self.gp.ZonalStatistics_sa(rgrp, "value", aspSinRad, y, "MEAN", "DATA")
        self.gp.ZonalStatistics_sa(zoneGD, "value", self.aspSinRad, y, "MEAN", "DATA")

        rx = self.rds_folder + "/" + "rx"
        self.cleanup(rx)
        self.gp.Power_sa(x, 2, rx)

        ry = self.rds_folder + "/" + "ry"
        self.cleanup(ry)
        self.gp.Power_sa(y, 2, ry)

        rs = self.rds_folder + "/" + "rs"
```

```
        self.cleanup(rs)
        self.gp.Plus_sa(rx,ry, rs)

        r = self.rds_folder + "/" + "r"
        self.cleanup(r)
        self.gp.SquareRoot_sa(rs, r)

        xr = self.rds_folder + "/" + "xr"
        self.cleanup(xr)
        self.gp.Divide_sa(x, r, xr)

        yr = self.rds_folder + "/" + "yr"
        self.cleanup(yr)
        self.gp.Divide_sa(y, r, yr)

        acosRad = self.rds_folder + "/" + "acosRad"
        self.cleanup(acosRad)
        self.gp.ACos_sa(xr, acosRad)

        acosDeg = self.rds_folder + "/" + "acosDeg"
        self.cleanup(acosDeg)
        self.gp.Times_sa(acosRad,57.2957795,acosDeg)

        avgDeg = self.rds_folder + "/" + "avgDeg"
        self.cleanup(avgDeg)
        condition = "CON (" + xr + " > 0 AND " + yr + " > 0, " + acosDeg + ", " + yr + " > 0 AND " + xr + "< 0, " +
acosDeg + ", " + xr + "< 0 AND " + yr + " < 0, 360 - " + acosDeg + ", 360 - " + acosDeg + ")"
        self.gp.SingleOutputMapAlgebra_sa(condition,avgDeg)

        MeanAspM90 = self.rds_folder + "/" + "MeanAspM90"
        self.cleanup(MeanAspM90)
        self.gp.Minus_sa(avgDeg,"90",MeanAspM90)

        MeanAspP90 = self.rds_folder + "/" + "MeanAspP90"
        self.cleanup(MeanAspP90)
        self.gp.Plus_sa(avgDeg,"90",MeanAspP90)

        OpLT180 = self.rds_folder + "/" + "OpLT180"
        self.cleanup(OpLT180)
        condition = "CON(" + self.Aspect + " < " + MeanAspM90 + " OR " + self.Aspect + " > " + MeanAspP90 + ",
1, 0)"
        self.gp.SingleOutputMapAlgebra_sa(condition,OpLT180)

        AspRangeGT180 = self.rds_folder + "/" + "AspRangeGT180"
        self.cleanup(AspRangeGT180)
        self.gp.Con_sa(AspectRange,self.Aspect,AspRangeGT180,"","VALUE > 180")

        AspShift360 = self.rds_folder + "/" + "AspShift360"
        self.cleanup(AspShift360)
        self.gp.Plus_sa(AspRangeGT180,"360", AspShift360)

        MeanAspP270 = self.rds_folder + "/" + "MeanAspP270"
        self.cleanup(MeanAspP270)
        self.gp.Plus_sa(avgDeg,"270",MeanAspP270)

        OpGE180 = self.rds_folder + "/" + "OpGE180"
        self.cleanup(OpGE180)
        condition = "CON(" + AspShift360 + " < " + MeanAspP90 + " OR " + AspShift360 + " > " + MeanAspP270 +
", 0, 1)"
        self.gp.SingleOutputMapAlgebra_sa(condition, OpGE180)

        OppSlope = self.rds_folder + "/" + "OppSlope"
        self.cleanup(OppSlope)
        self.gp.Con_sa(AspectRange,OpLT180,OppSlope,OpGE180,"VALUE < 180")

        # ALLOCATE MEANDIRECTION of MAIN SLOPE to OPPOSITESLOPE
        MainSlp = self.rds_folder + "/" + "MainSlp"
        self.cleanup(MainSlp)
        self.gp.SetNull_sa(OppSlope, zoneGD, MainSlp, "value = 1")

        #Calc MeanDir in MainSlp
        MMeanSlpDir = self.rds_folder + "/" + "MMeanSlpDir"
        self.cleanup(MMeanSlpDir)
        self.gp.ZonalStatistics_sa(MainSlp, "Value", self.mainSlpDir, MMeanSlpDir, "Mean", "DATA")

        MMeanSlpDir05 = self.rds_folder + "/" + "MMeanSlpDir05"
        self.cleanup(MMeanSlpDir05)
```

```
        self.gp.Plus_sa(MMeanSlpDir,"0.5",MMeanSlpDir05)

        iMMeanSlpDir = self.rds_folder + "/" + "iMMeanSlpDir"
        self.cleanup(iMMeanSlpDir)
        self.gp.int_sa(MMeanSlpDir05,iMMeanSlpDir)

        MainMeanDir = self.rds_folder + "/" + "MainMeanDir"
        self.cleanup(MainMeanDir)
        reclassifyRanges = "0 1.5 1;1.5 3 2;3 6 4;6 12 8;12 24 16;24 48 32; 48 96 64;96 192 128"
        self.gp.Reclassify_sa(iMMeanSlpDir, "Value", reclassifyRanges, MainMeanDir, "DATA")

        #oppSlp NoData, allocate value = 1, Main Value = 0
        MainOpp = self.rds_folder + "/" + "MainOpp"
        self.cleanup(MainOpp)
        self.gp.IsNull_sa(MainSlp, MainOpp)

        NibDir = self.rds_folder + "/" + "NibDir"
        self.cleanup(NibDir)
        self.gp.Con_sa(MainOpp, MainMeanDir, NibDir, MainOpp,"value = 0")

        oppSlpMainDir = self.rds_folder + "/" + "oppSlpMainDir"
        self.cleanup(oppSlpMainDir)
        self.gp.Nibble_sa(NibDir, MainMeanDir, oppSlpMainDir)

        #if not OppSlope, then mainSlpDir, else oppSlpMainDir
        slopeDir = self.rds_folder + "/" + "slopeDir"
        self.cleanup(slopeDir)
        self.gp.Con_sa(OppSlope, self.mainSlpDir, slopeDir, oppSlpMainDir,"value = 0")

        #Determine flowdirection
        pressDir = self.rds_folder + "/" + "pressDir"
        self.cleanup(pressDir)
        self.gp.FlowDirection_sa(xTinRaster,pressDir)

        directionGd = self.rds_folder + "/" + "directionGd"
        self.cleanup(directionGd)
        condition = "CON(" + zoneGD + " > 0, CON( " + self.Aspect + " < 0 OR " + self.Slope + " < 5," + pressDir +
"," + slopeDir + "))"
        self.gp.SingleOutputMapAlgebra_sa(condition, directionGd)

        #----------Mask----------
        self.gp.Mask = zoneGD

        scriptmsg = "expand, ws " + nameid_base
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)

        #----------EXPAND srcFC_ph for ws-------------------------------------
        SPgd0 = self.rds_folder + "/" + "SPgd0"
        self.cleanup(SPgd0)
        self.gp.Con_sa(self.SPgd,"88",SPgd0,"","VALUE > -1")

        try:
            SPgdExpand = self.rds_folder + "/" + "SPgdExpand"
            self.cleanup(SPgdExpand)
            self.gp.Expand_sa(SPgd0,SPgdExpand,"1","88")

            wList = [self.rds_folder + "/" + 'ws']
            eList = [self.rds_folder + "/" + 'exp']

            cList = range(5)
            wcList = [(a+str(b)) for a in wList for b in cList]
            ecList = [(a+str(b)) for a in eList for b in cList]

            ecList[0] = (SPgdExpand)
            ecList.append(self.rds_folder + "/" + 'exp5')

            for i in cList:
                self.gp.Watershed_sa(directionGd, ecList[i], wcList[i])
                self.gp.Expand_sa(wcList[i], ecList[i+1], "1", "88")

            actLayer = self.tar_gdb + "/" + nameid_base + "_ws"
            self.cleanup(actLayer)

            self.gp.RasterToPolygon_conversion(wcList[4], actLayer, "simplify")
            scriptmsg = "Einzugsgebiete zu " + self.nameid + " berechnet: " + os.path.basename(actLayer)
            pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
```

```
    except:
        scriptmsg = "Es liegen keine Zellen des Schadenpotenzials-Rasters innerhalb "+ Expression + "!"
        pp_util_GIS.gpmsg(self.gp,scriptmsg,1)

    self.cleanup(xoutTin)
    self.cleanup(xTinRaster)
    self.cleanup(fc_ph)
    self.cleanup(fc_cap)


    return
# ---------------------------------------------------------------------------
def GET_NAMEIDS(self):

    # Liste der ph Features fuer die Prozesssteuerung
    nameid_list = []
    rows = self.gp.SearchCursor(self.srcFC_ph)
    row = rows.Next()
    while row:
        # Eintraege Prozessart auslesen
        nameid = row.GetValue("nameid")
        if nameid not in nameid_list:
            nameid_list.append(nameid)
        row = rows.Next()
    del row, rows

    S_file = open(self.nameidfile, "w")
    for nameid in nameid_list:
        S_file.write(nameid + "\n")
    S_file.close()

    return

# ---------------------------------------------------------------------------
def RDS_BASIS_GEBIET(self):
    scriptmsg = "berechne Rastergrundlagen zum Extent von " + os.path.basename(self.srcFC_ph)
    pp_util_GIS.gpmsg(self.gp,scriptmsg,0)


    # Extent festlegen
    self.gp.extent = self.setextent(self.srcFC_ph)
    self.gp.snapraster = self.dtm


    self.cleanup(self.Aspect)
    self.gp.Aspect_sa(self.dtm, self.Aspect)

    self.cleanup(self.aspRad)
    self.gp.Times_sa(self.Aspect,0.017453293,self.aspRad)

    self.cleanup(self.aspCosRad)
    self.gp.Cos_sa(self.aspRad,self.aspCosRad)

    self.cleanup(self.aspSinRad)
    self.gp.Sin_sa(self.aspRad,self.aspSinRad)

    self.cleanup(self.Slope)
    self.gp.Slope_sa(self.dtm,self.Slope)

    self.cleanup(self.mainSlpDir)
    self.gp.FlowDirection_sa(self.dtm,self.mainSlpDir)

    self.cleanup(self.SPgd)
    self.gp.FeatureToRaster_conversion(self.srcFC_spot,"Shape_Area",self.SPgd,self.Aspect)

    self.cleanup(self.aspRad)

    return



#---------------------------------------------------------------------------
def PROC_SPOTREL_PH(self):
    # ------> Auswertung schadenpotenzialsrelevante ph (Prozessraeume) zu LARGE

    self.pp_proc_spotrel_ph()
```

```
        if not self.gp.exists(self.srcFC_ph):
            scriptmsg = "Abbruch! Grundlage fehlt: " + self.srcFC_ph
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()

        if not self.gp.exists(self.srcFC_rel):
            scriptmsg = "Abbruch! Grundlage fehlt: " + self.srcFC_rel
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()

        return



    #---------------------------------------------------------------------------
    def BASISDATEN(self):
        # ------> Bereitstellung der zu verarbeitenden Datenstrukturen zum Gebiet
        # ------> Die Datenstrukturen in src_Workspace sind Coverages

        self.gp.workspace = self.src_Workspace
        self.gp.scratchworkspace = self.src_Workspace

        # cap-Datenquelle in tar_gdb kopieren (Punkt-Angaben zu Lawinen-Drucken)
        scriptmsg = "berechne " + os.path.basename(self.srcFC_cap)
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
        srcFC_cap = self.pp_proc_cap()
        self.cleanup(self.srcFC_cap)
        self.gp.CopyFeatures(srcFC_cap,self.srcFC_cap)

        # Schadenpotenzial in tar_gdb importieren,
        scriptmsg = "berechne " + os.path.basename(self.srcFC_spot)
        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
        expression = ""
        self.cleanup(self.srcFC_spot)
        self.gp.FeatureClassToFeatureClass_conversion(self.DamLayer,                                    self.tar_gdb,
os.path.basename(self.srcFC_spot), expression)



    #---------------------------------------------------------------------------
    def CREATE_WORKSPACES(self):
        # ------> self.tar_gdb erstellen
        try:
            self.cleanup(self.tar_gdb)
            self.gp.CreateFileGDB_management(self.tar_Folder, self.tar_gdbname)

            if self.gp.exists(self.tar_gdb):
                scriptmsg = "Ablage Ergebnisse in: " + self.tar_gdb
                pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
        except:
            scriptmsg = "Abbruch: " + self.tar_gdbname + " kann in " + self.tar_Folder + " nicht erstellt werden!"
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()

        # ------> self.rds_folder fuer TIN und Raster-Zwischenergebnisse erstellen
        if not os.path.isdir(self.rds_folder):
            try:
                os.mkdir(self.rds_folder)
            except:
                scriptmsg = "Abbruch: " + os.path.basename(self.rds_folder) + " kann in " + self.tar_Folder + " nicht
erstellt werden!"
                pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
                sys.exit()
        return


    #---------------------------------------------------------------------------
    def cleanup(self,infc):
        if self.gp.exists(infc):
            self.gp.delete(infc)
        return

    # ---------------------------------------------------------------------------
    def pp_proc_cap(self):

        mycapFClist = []
```

```
        self.gp.workspace = self.src_Workspace
        searchstring = self.gebiet.lower() + "n*_cap"
        myWSlist = self.gp.ListWorkspaces(searchstring)

        if len(myWSlist)>0:

            for mycapFDS in myWSlist:
                self.gp.workspace = mycapFDS
                mycapFCs = self.gp.ListFeatureClasses("*", "Point")
                for srcFC_cap in mycapFCs:
                    #print os.path.basename(srcFC_cap)
                    mycapFClist.append(srcFC_cap)

        if len(mycapFClist)<>1:
            scriptmsg = "Abbruch! Zu gebiet " + self.gebiet + " sind mehrere/keine cap Datenquellen definiert!"
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()


        return srcFC_cap

    # ---------------------------------------------------------------------------
    def bildeFC_Paare_ph_rel(self,outFCs):

        # FC Liste filtern auf Feature Klassen mit Gebietsname und ph_string oder rel_string
        theFCs = []
        for myFC in outFCs:
            myfdsname = os.path.dirname(myFC)
            mydirname = os.path.dirname(myfdsname)

            afcname = string.replace(myFC.lower(), mydirname.lower(), "")
            for ffix in [self.ph_string, self.rel_string]:
                if afcname.find(self.gebiet) >= 0:
                    if afcname.find(ffix) >= 0:
                        theFCs.append(myFC)


        # FC Liste filtern auf Feature Klassen mit Feldern nameid
        keepFCs = []
        fieldlist = ["nameid", "id"]
        for myFC in theFCs:
            anzfound = 0
            for searchfieldname in fieldlist:

                if len(self.gp.ListFields(myFC,searchfieldname))>0:
                    anzfound = anzfound + 1

            if anzfound == len(fieldlist):
                keepFCs.append(myFC)


        # ph und rel FC Listen aufbauen
        ph_list = []
        rel_list = []
        for myFC in keepFCs:
            myfdsname = os.path.dirname(myFC)
            mydirname = os.path.dirname(myfdsname)

            afcname = string.replace(myFC.lower(), mydirname.lower(), "")
            if afcname.find(self.ph_string) >= 0:
                ph_list.append(myFC)
            elif afcname.find(self.rel_string) >= 0:
                rel_list.append(myFC)

        # zusammengehoerige Paare bilden
        srcPairs = []
        for phFC in ph_list:
            afcnameph = string.replace(phFC.lower(), self.ph_string.lower(), "")
            #print afcnameph
            for relFC in rel_list:
                afcnamerel = string.replace(relFC.lower(), self.rel_string.lower(), "")
                #print afcnamerel
                if (afcnameph == afcnamerel):
                    srcPairs.append([phFC,relFC])

        anzpairs = len(srcPairs)
```

```
    scriptmsg = str(len(srcPairs)) + " gueltige ph / rel FC Paare zu Gebiet " + self.gebiet
    pp_util_GIS.gpmsg(self.gp,scriptmsg,0)


    return srcPairs


# ---------------------------------------------------------------------------
def procFDSlist(self,myFDSlist):

    outFCs = []
    myFClist = []
    for myFDS in myFDSlist:

        self.gp.workspace = myFDS
        myFCs = self.gp.ListFeatureClasses("*.*", "Region")
        for myFC in myFCs:
            FullFC = myFDS + "/" + myFC
            if self.gp.exists(FullFC):
                myFClist.append(FullFC)


    if len(myFClist)>0:

        for fFC in myFClist:
            dsc = self.gp.describe(fFC)
            # wenn's eine FAT hat, ists ein Coverage
            if dsc.HasFAT:
                if dsc.FeatureClassType == "Region":
                    outFCs.append(fFC)
            # kein Coverage
            else:
                if dsc.FeatureClassType == "Region":
                    outFCs.append(fFC)

    return outFCs


# ---------------------------------------------------------------------------
def pp_proc_spotrel_ph(self):


    self.gp.workspace = self.src_Workspace
    searchstring = self.gebiet.lower() + "n*"
    myWSlist = self.gp.ListWorkspaces(searchstring)
    myFDSlist = []
    for entry in myWSlist:
        self.gp.workspace = entry
        myFClist = self.gp.ListFeatureClasses("*","REGION")
        if len(myFClist)>0:
            myFDSlist.append(entry)

    if len(myFDSlist)==0:
        scriptmsg = "Abbruch: keine gueltigen Regions in " + self.src_Workspace
        pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
        sys.exit()


    # FC Liste erstellen

    srcPairs = []

    outFCs = self.procFDSlist(myFDSlist)

    # self.gebietsweise ph und rel FC Paare pruefen.
    if len(outFCs) > 0:
        srcPairs = self.bildeFC_Paare_ph_rel(outFCs)


        self.gp.ScratchWorkspace = self.tar_gdb
        self.gp.workspace = self.tar_gdb



        # wenn self.gebietsweise ph und rel Paare vorhanden...
        if len(srcPairs) > 0:
```

```
# ph's zusammenfuehren in Ziel Feature Klasse in tar_Folder
# beinhaltet alle Features ph, bewaldet und waldfrei
expression = ""
phFC = ""
for pair in srcPairs:
    phFC = pair[0]
    if len(expression) == 0:
        expression = phFC
    else:
        expression = expression + ";" + phFC
tar_PHall = "x_" + self.ph_string + "_all"
self.cleanup(tar_PHall)
self.gp.Merge_management(expression, tar_PHall, "NO_TEST")


# von den phs wird nur range_code 2 und 4 verwendet.
# code 0 liegt ausserhalb des Lawinenperimeters
tar_PH = "x_phc"
self.cleanup(tar_PH)
selexpr = "range_code" + " <> " + "0"
self.gp.Select_analysis(tar_PHall, tar_PH, selexpr)
# dissolve der Lawinenzuege auf nameid
tar_phFC = "x_" + self.ph_string
self.cleanup(tar_phFC)
self.gp.Dissolve_management(tar_PH, tar_phFC, "nameid", "", "SINGLE_PART")
self.cleanup(tar_PHall)
self.cleanup(tar_PH)


# rel's zusammenfuehren in Ziel Feature Klasse in tar_Folder
# beinhaltet alle Features rel, bewaldet und waldfrei
expression = ""
for pair in srcPairs:
    relFC = pair[1]
    if len(expression) == 0:
        expression = relFC
    else:
        expression = expression + ";" + relFC
tar_relFC = "x_" + self.rel_string
self.cleanup(tar_relFC)
self.gp.Merge_management(expression, tar_relFC, "NO_TEST")


#Make Feature Layer Damage
MemDamage = "MemDamage"
self.cleanup(MemDamage)
self.gp.MakeFeatureLayer(self.srcFC_spot,MemDamage)

for size in ["s", "m", "l"]:
    scriptmsg = "verarbeite Features Kategorie: " + size + ", gebiet: " + self.gebiet
    pp_util_GIS.gpmsg(self.gp,scriptmsg,0)

    sizetext = ""

    if size == "s":
        sizetext = "small"
    if size == "m":
        sizetext = "medium"
    if size == "l":
        sizetext = "large"



    ph_namelist = []
    rel_namelist = []



    # ---------- VERARBEITUNG Schadenpotenzialsrelevanz PH
    # ph_namelist fuer Groessenkategorie erstellen,
    # wegfiltern Eintraege n (Nichtwald)


    # Featurelayer ph anlegen
    MemPH = "MemPH"
    self.cleanup(MemPH)
    self.gp.MakeFeatureLayer(tar_phFC,MemPH)
```

```
                # Featurelayer rel anlegen
                MemREL = "MemREL"
                self.cleanup(MemREL)
                self.gp.MakeFeatureLayer(tar_relFC,MemREL)


                # Selektion der Features n, self.gebiet und Groessenklasse
                anzPHsize = 0
                anzPHspot = 0
                selstring = self.gebiet + self.wald_string + "_" + size + "%"
                Exp_ph = "NAMEID LIKE '" + selstring + "'"
                scriptmsg = "Selektiere PH " + selstring
                pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
                self.gp.SelectLayerByAttribute(MemPH,"NEW_SELECTION",Exp_ph)
                obj_count = self.gp.GetCount_management(MemPH)
                anzPHsize = int(obj_count.GetOutput(0))


                # Reduktion auf schadenpotenzialsrelevante ph Flaechen
                if anzPHsize > 0:
                    scriptmsg = "Selektiere PH spot relevante"
                    pp_util_GIS.gpmsg(self.gp,scriptmsg,0)

self.gp.SelectLayerByLocation_Management(MemPH,"INTERSECT",MemDamage,"","SUBSET_SELECTION")
                    obj_count = self.gp.GetCount_management(MemPH)
                    anzPHspot = int(obj_count.GetOutput(0))
                    scriptmsg = "in PH " + self.wald_string + ", Gebiet " + self.gebiet + ", " + size + ": " +
str(anzPHsize) + " Features"
                    pp_util_GIS.gpmsg(self.gp,scriptmsg,0)


                    # Resultat pro Groessenklasse schreiben
                    if anzPHspot > 0:
                        nameout = self.gebiet + self.wald_string + "_" + self.ph_string + "_" + sizetext
                        expression = ""
                        outFCph = self.tar_gdb + "/" + nameout
                        self.cleanup(outFCph)
                        self.gp.FeatureClassToFeatureClass_conversion(MemPH, self.tar_gdb, nameout, expression)
                        scriptmsg = "in PH " + self.wald_string + ", Gebiet " + self.gebiet + ", " + size + ": " +
str(anzPHsize) + " Features spot relevant"
                        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)



                # ---------- VERARBEITUNG REL
                # Die spot relevanten PH Features werden mit den rel Features verschnitten.
                if anzPHspot > 0:
                    nameout = "xi_" + self.rel_string
                    tmpFCrel = self.tar_gdb + "/" + nameout
                    self.cleanup(tmpFCrel)
                    intersectstring = tar_relFC + ";" + outFCph
                    self.gp.Intersect_analysis(intersectstring, tmpFCrel)
                    self.gp.AddField_management(tmpFCrel,
"phOK","SHORT","","","","","NULLABLE","REQUIRED","")

                # In einem update cursor werden die rel FlÃ¤chen in den korrekten ph FlÃ¤chen markiert
                    rows = self.gp.UpdateCursor(tmpFCrel)
                    row = rows.Next()


                    while row:
                        # Eintraege Prozessart auslesen
                        nid1 = row.GetValue("nameid")
                        nid2 = row.GetValue("nameid_1")
                        name1 = string.replace(nid1.lower(), self.rel_string.lower(), "")
                        name2 = string.replace(nid2.lower(), self.ph_string.lower(), "")
                        if name1 == name2:
                            row.phOK = "1"

                        rows.UpdateRow(row)
                        row = rows.Next()
                    del row, rows


                    # Reduktion auf zulaessige rel FlÃ¤chen
```

```
                    nameout = "x" + self.gebiet + self.wald_string + "_" + self.rel_string + "_" + sizetext
                    outFCrelOK = self.tar_gdb + "/" + nameout
                    self.cleanup(outFCrelOK)
                    selexpr = "phOK" + " = " + "1"
                    self.gp.Select_analysis(tmpFCrel, outFCrelOK, selexpr)


                    obj_count = self.gp.GetCount_management(outFCrelOK)
                    anzREL = int(obj_count.GetOutput(0))
                    if anzREL > 0:
                        scriptmsg = "in zugehoerigen REL " + self.wald_string + ", Gebiet " + self.gebiet + ", " + size +
": " + str(anzPHsize) + " Features"
                        pp_util_GIS.gpmsg(self.gp,scriptmsg,0)

                        self.cleanup(tmpFCrel)
                        nameout = self.gebiet + self.wald_string + "_" + self.rel_string + "_" + sizetext
                        outFCrel = self.tar_gdb + "/" + nameout
                        self.cleanup(outFCrel)
                        self.gp.Dissolve_management(outFCrelOK, outFCrel, "nameid", "", "SINGLE_PART")
                        self.cleanup(outFCrelOK)


                self.cleanup(tar_phFC)
                self.cleanup(tar_relFC)



            else:
                scriptmsg = "Abbruch: Keine gueltigen Daten zu gebiet " + self.gebiet
                pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
                sys.exit()

        else:
            scriptmsg = "Abbruch: Keine Feature Klassen in " + self.src_Workspace
            pp_util_GIS.gpmsg(self.gp,scriptmsg,2)
            sys.exit()


        return


    # ---------------------------------------------------------------------------
    def setextent(self,myextfc):
        dsc = self.gp.describe(myextfc)
        myextent = dsc.Extent

        xmin = myextent.xmin - pp_Settings.addext
        ymin = myextent.ymin - pp_Settings.addext
        xmax = myextent.xmax + pp_Settings.addext
        ymax = myextent.ymax + pp_Settings.addext


        myextent = str(xmin) + " " + str(ymin) + " " + str(xmax) + " " + str(ymax)

        return myextent


    '''
    scriptmsg = "Dauer der Verarbeitung: " + timeconsumstring(starttime, time.clock())
    pp_util_GIS.gpmsg(self.gp,scriptmsg,0)
    '''
```

9.3.4 Pp_Settings.py: Parametrisierung

```
ï»¿'''
pp_Settings.py

Der Prozess pp_large setzt die VerfÃ¼gbarkeit folgender Skripts voraus:
pp_large_launch.py (aufrufendes Skript)
pp_large_proc_nameid.py
pp_large_proc_rel.py
pp_Settings.py
pp_util_GIS.py

Autor:      geo7 AG, Peter Gsteiger

History:    30.06.2011 programmiert fuer ArcGIS 9.3.1, Python 25

Version:    1.0


Description:
Konfiguration Prozess pp_large

Workflow und Produkte sind beschrieben in:
Identifikation schadenpotenzialsrelevanter Lawinenanrissgebiete Silvaprotect
geo7, 30.06.2011


'''




# ---------------------------------------------------------------------------
# Python Modules
# ---------------------------------------------------------------------------
import sys, os

# ---------------------------------------------------------------------------
# Settings
# ---------------------------------------------------------------------------

# erforderliche ArcGIS-Lizenzierung:
ARCLICREQ = "ArcInfo"

# Parameter:
src_Workspace = r"D:\SilvaProtect\BASIC\INPUT\WIL\loadFC"
tar_Folder = r"D:\SilvaProtect\SZENARIEN\SBB\IN_urGN_Lawine"
gebiet = "WIL"
DamLayer = r"D:\SilvaProtect\SZENARIEN\SBB\DA_urGN_Lawine\DA_wil.shp"
dtm = r"D:\SilvaProtect\BASIC\DHM.mdb\DHM_10"


# Installationspfad Python:
_path_python = r'"C:\Program Files (x86)\Python25"'
_path_python = r"C:\Python25"
# Pfad Skripte:
_path_script = r"D:\SilvaProtect\INTERSECT\LAWINE_geo7\20110704_BAFU\Scripts"



# Zuschlag [m] Mapextent bei Rasterverarbeitungen
addext = 500




# Variabeln zum nameid handling
# !!!!!Eintrage nicht veraendern!!!!!
ph_string = "ph"
rel_string = "rel"
cap_string = "cap"
wald_string = "n"
```

9.3.5 Pp_util_GIS.py: GIS Anfrage und Warnemungen

```
ï»¿'''
pp_util_GIS.py

Der Prozess pp_large setzt die VerfÃ¼gbarkeit folgender Skripts voraus:
pp_large_launch.py (aufrufendes Skript)
pp_large_proc_nameid.py
pp_large_proc_rel.py
pp_Settings.py
pp_util_GIS.py

Autor:      geo7 AG, Peter Gsteiger

History:    30.06.2011 programmiert fuer ArcGIS 9.3.1, Python 25

Version:    1.0


Description:
Bezug gp-Objekt, Einstellungen zur Geoverarbeitung, Benachrichtigungen

Workflow und Produkte sind beschrieben in:
Identifikation schadenpotenzialsrelevanter Lawinenanrissgebiete Silvaprotect
geo7, 30.06.2011


'''


# -----------------------------------------------------------------------------------
# Python Modules
# -----------------------------------------------------------------------------------
import sys, os, string,time


# -----------------------------------------------------------------------------------
def gdbsettings(gp):

    gp.outputCoordinateSystem = "PROJCS['CH1903_LV03',GEOGCS['GCS_CH1903',DATUM['D_CH1903',SPHEROID['Bessel_1841',6377397.155,299.1528128]],PRIMEM['Greenwich',0.0],UNIT['Degree',0.0174532925199433]],PROJECTION['Hotine_Oblique_Mercator_Azimuth_Center'],PARAMETER['False_Easting',600000.0],PARAMETER['False_Northing',200000.0],PARAMETER['Scale_Factor',1.0],PARAMETER['Azimuth',90.0],PARAMETER['Longitude_Of_Center',7.439583333333333],PARAMETER['Latitude_Of_Center',46.95240555555556],UNIT['Meter',1.0]]"

    gp.XYResolution = "0.001 Meters"
    gp.MResolution = "0.001"
    gp.ZResolution = "0.001 Meters"
    gp.XYTolerance = "0.002 Meters"
    gp.MTolerance = "0.002"
    gp.ZTolerance = "0.002 Meters"

    gp.randomGenerator = "0 ACM599"
    gp.outputZFlag = "Same As Input"
    gp.outputMFlag = "Same As Input"
    gp.outputZValue = ""
    gp.qualifiedFieldNames = "true"
    gp.extent = "DEFAULT"
    gp.geographicTransformations = ""

    return

# -----------------------------------------------------------------------------------
def GIS_Setup(ARCLICREQ):

    gp = None
    scriptmsg = ""

    # Geoprocessor Objekt erstellen
    try:
        import arcgisscripting
        gp = arcgisscripting.create(9.3)
    except:
        scriptmsg = "Abbruch: Das Geoprocessor Objekt konnte nicht erstellt werden!"
        gpmsg(gp,scriptmsg,2)
        sys.exit()
```

```
    # Liste alle Lizenzen
    licenses_all = []
    licenses_all.append("ArcEditor")
    licenses_all.append("ArcView")
    licenses_all.append("ArcInfo")


    # bestimmen verfÃ¼gbare Lizenzen
    licenses_available = []
    rsp_suit = ""
    for lic in licenses_all:
        rsp_suit = gp.CheckProduct(lic)
        #scriptmsg = "CheckProduct fuer " + lic + " hat Status " + str(rsp_suit)
        #gpmsg(gp,scriptmsg,0)
        if rsp_suit in ("Available","AlreadyInitialized"):
            licenses_available.append(lic)


    # bestimmen taugliche Lizenzen
    licenses_suitable = []
    if string.lower(ARCLICREQ) == string.lower("ArcView"):
        licenses_suitable.append("ArcView")
        licenses_suitable.append("ArcEditor")
        licenses_suitable.append("ArcInfo")
    elif string.lower(ARCLICREQ) == string.lower("ArcEditor"):
        licenses_suitable.append("ArcEditor")
        licenses_suitable.append("ArcInfo")
    elif string.lower(ARCLICREQ) == string.lower("ArcInfo"):
        licenses_suitable.append("ArcInfo")


    # bestimmen Schnittmenge verfÃ¼gbare und taugliche Lizenzen
    licenses_needed = []
    for lic_t in licenses_suitable:
        for lic_v in licenses_available:
            if lic_t == lic_v:
                licenses_needed.append(lic_t)
    if len(licenses_needed) == 0:
        scriptmsg = "Es ist keine Lizenz " + ARCLICREQ + " verfuegbar. "
        gpmsg(gp,scriptmsg,2)
        sys.exit()



    # aktive Lizenz (Desktop Manager) bestimmen
    try:
        gp.SetProduct(ARCLICREQ)
    except:
        scriptmsg = "Abbruch! Desktop Manager Lizenz korrigieren auf " + str(ARCLICREQ)
        gpmsg(gp,scriptmsg,2)
        sys.exit()


    # Load required Extensions...
    try:
        gp.CheckOutExtension("spatial")
    except:
        scriptmsg = "Abbruch: Die Spatial Extension ist nicht verfuegbar."
        gpmsg(gp,scriptmsg,2)
        sys.exit()


    # Load required Extensions...
    try:
        gp.CheckOutExtension("3D")
    except:
        scriptmsg = "Abbruch: Die 3D Extension ist nicht verfuegbar."
        gpmsg(gp,scriptmsg,2)
        sys.exit()

    return gp

# ---------------------------------------------------------------------------
def gpmsg(gp,message,sev):

    yr = time.strftime("%Y")
```

```
mt = time.strftime("%m")
d = time.strftime("%d")
hr = time.strftime("%H")
mte  = time.strftime("%M")
sec  = time.strftime("%S")
timestring =  yr + "_" + mt + "_" + d + " " + hr + ":" + mte + ":" + sec

message = str(timestring) + " " + message


try:
    if sev == 2:
        gp.AddError(message)
    elif sev == 1:
        gp.AddWarning(message)
    else:
        gp.AddMessage(message)
    print message
except:
    print message


return
```

9.4 Berechungszeit

Modul	Geschätzte Zeit
DAMAGE	5 h
Lawine (INTERSECT)	100 h
Sturz (INTERSECT)	30 h
Hangmure (INTERSECT)	50 h
Relevante Gerinne (INTERSECT)	40 h
Gerinneprozesse (INTERSECT)	150 h
SWI und SPI (SYNTHESE)	5 h
Geschätzer Gesamtaufwand	**380 h**