Entwicklung eines modernen GUI für INTERLIS Tools

Erneuerte Benutzeroberfläche für das INTERLIS Tool ‘ilivalidator’

Situation INTERLIS

Bestehende INTERLIS Tools sind für ein Fachpublikum entwickelt. Das Graphische User Interface (GUI) steht nicht im Vordergrund.

INTERLIS Validator als Tool täglich im Gebrauch.

Hauptziel der Thesis:
Die Entwicklung einer verständlichen GUI anhand Grundprinzipien des Interface-Design.

Leitfragen:
- *Was für Design- und Softwareaspekte sind beim Aufbau eines GUIs zu beachten?*
- *Wie kann eine User-Experience für verschiedene Endnutzer auf ein GUI eingepasst werden?*
- *Wie kann eine Software portabel gestaltet werden?*

Design-Konzept

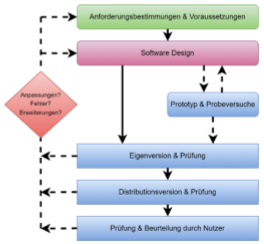
Leitziele

Konkrete Leitziele für die Entwicklung nach den goldenen Regeln des Interface-Designs von Chandra & Guntupalli (2008)

Leitziel 1	Der Validierungsablauf erfolgt immer auf identische Weise.
Leitziel 2	Drag und Drop Funktionalität wird implementiert. Einstellungen werden visuell in das Interface verbaut.
Leitziel 3	Rückmeldungen werden über GitHub gesammelt.
Leitziel 4	Die Nutzer erhalten nur relevante Validierungsinformationen. Dialoge sind dynamisch ausklappbar zur Schlichthaltung der App.
Leitziel 5	Der Validierungsprozess kann nach Fehlerbehebungen einer Datei direkt weitergefahren werden.
Leitziel 6	Der Gesamtablauf kann zurückgesetzt werden.
Leitziel 7	Validierungs- und Darstellungsoptionen werden den Nutzern zur Verfügung gestellt.
Leitziel 8	Hilfehinweise erscheinen mit Hover- bzw. Schwebeeffekte.

Iterationsprozess

Fest definierte Iterationsschritte für die Entwicklung



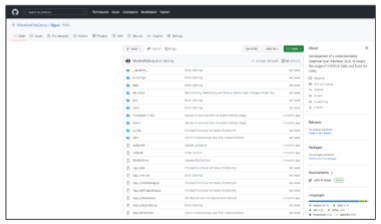
Adaptiert von Blair-Early und Zender (2008)

Software-Architektur

- Erstellen des GUI über den bestehenden ilivalidator. Die interne Software wird nicht verändert.
- Arbeit mit Python und PyQt6. Mit der Qt Designer Software bleibt das GUI auch zukünftig dynamisch anpassbar.
- Einbau der jeweils neusten ilivalidator Version und Java Runtime Environment.
- Distribution über PyInstaller Bibliothek auf dem jeweils erwünschten Betriebssystem.

Distribution und Aktualisierung

- Projekt Homepage über GitHub Pages veröffentlicht.
- Portables Executable für Windows und Unix zum Download verfügbar. MacOS folgt bei Nutzerbedarf.
- Gesamter Quellcode dokumentiert und offen auf GitHub unter ‘iligui’ erhältlich.
- Aktualisierungen werden auf der Homepage mitgeteilt und neu distribuiert.

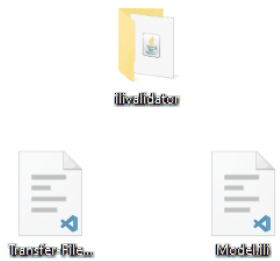


Zukunft des iligui

Hilfehinweise für Fehler werden laufend ergänzt.
Das Projekt wird auf dem INTERLIS Forum publiziert.

Mögliche Weiterentwicklung:

- Übersetzungen auf Deutsch, Französisch, Spanisch
- Validierungen von mehreren Transferfiles im GUI
- Einstellungen via CONFIG.ini Files
- Einbau des INTERLIS Model Browser
- Zusätzliche Lösungsbeispiele zu den Fehlerhinweisen im GUI
- Automatisiertes Deployment über GitHub



Situation INTERLIS

Bestehende INTERLIS Tools sind für ein Fachpublikum entwickelt. Das Graphische User Interface (GUI) steht nicht im Vordergrund.

INTERLIS Validator als Tool täglich im Gebrauch.

Hauptziel der Thesis:
Die Entwicklung einer verständlichen GUI anhand Grundprinzipien des Interface-Design.

Leitfragen:

- *Was für Design- und Softwareaspekte sind beim Aufbau eines GUIs zu beachten?*
- *Wie kann eine User-Experience für verschiedene Endnutzer auf ein GUI eingepasst werden?*
- *Wie kann eine Software portabel gestaltet werden?*

Aufbau INTERLIS

Beschreibungssprache für Geodaten bestehend aus zwei Elementen:

- Die Modelldatei mit der Dateiendung '.ili'.
- Die Transferdatei mit der Dateiendung '.itf' / '.xtf'.

INTERLIS-2 MODELL

```
INTERLIS 2.3;
MODEL RoadsSimple (en) AT "http://www.interlis.ch/models"
  VERSION "2016-08-11" =
    UNIT
      Angle_Degree = 180 / PI [INTERLIS.rad];
    DOMAIN
      Point2D = COORD
        0.000 .. 200.000 [INTERLIS.m], !! Min_East  Max_East
        0.000 .. 200.000 [INTERLIS.m], !! Min_North Max_North
        ROTATION 2 -> 1;
      Orientation = 0.0 .. 359.9 CIRCULAR [Angle_Degree];

  TOPIC Roads =
    CLASS LandCover =
      Type: MANDATORY (
        building,
        street,
        water,
        other);
      Geometry: MANDATORY SURFACE WITH (STRAIGHTS)
        VERTEX Point2D WITHOUT OVERLAPS > 0.100;
    END LandCover;
  END Roads; !! of TOPIC

END RoadsSimple. !! of MODEL
```

INTERLIS-2 TRANSFER-DATEI

```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSFER xmlns="http://www.interlis.ch/INTERLIS2.3">

<HEADERSECTION
  SENDER="ili2fme-6.1.2-20160811" VERSION="2.3">
  <MODELS><MODEL NAME="RoadsSimple" VERSION="2016-08-11"
  URI="http://www.interlis.ch/models"></MODEL></MODELS>
</HEADERSECTION>

<DATASECTION>
  <RoadsSimple.Roads BID="505">
     <RoadsSimple.Roads.LandCover TID="16">
        <Type>water</Type>
        <Geometry><SURFACE><BOUNDARY><POLYLINE>
        <COORD><C1>39.038</C1><C2>60.315</C2></COORD>
        <COORD><C1>41.2</C1><C2>59.302</C2></COORD>
       </POLYLINE></BOUNDARY></SURFACE></Geometry>
     </RoadsSimple.Roads.LandCover>

     <RoadsSimple.Roads.LandCover TID="18">
        <Type>building</Type>
        <Geometry><SURFACE><BOUNDARY><POLYLINE>
        <COORD><C1>101.459</C1><C2>65.485</C2></COORD>
        <COORD><C1>108.186</C1><C2>69.369</C2></COORD>
        </POLYLINE></BOUNDARY></SURFACE></Geometry>
     </RoadsSimple.Roads.LandCover>
  </RoadsSimple.Roads>
</DATASECTION>

</TRANSFER>
```

Design-Konzept

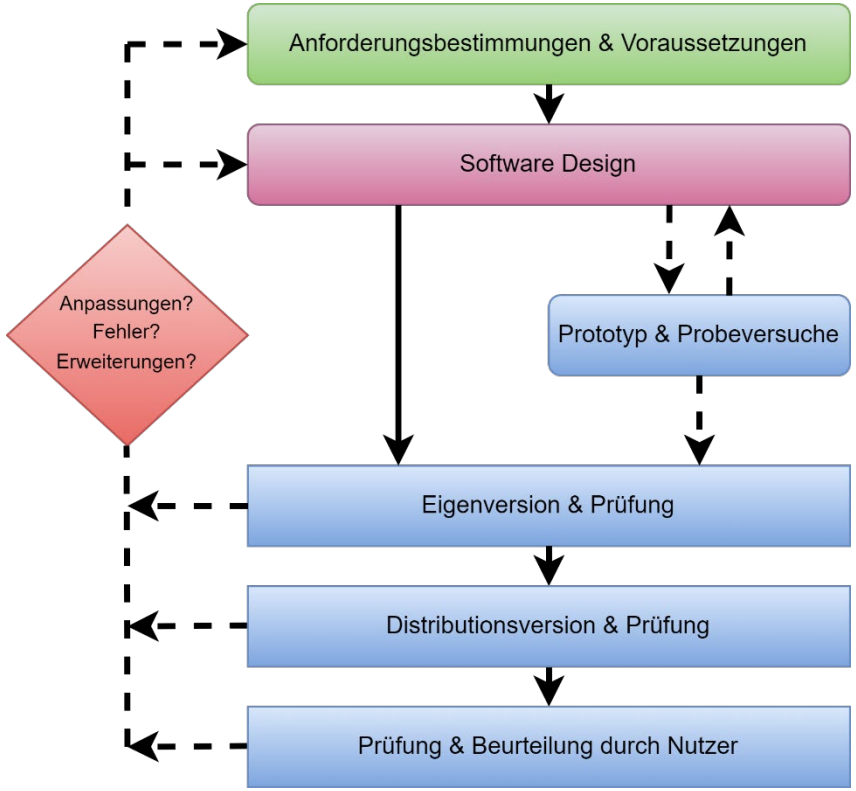
Leitziele

Konkrete Leitziele für die Entwicklung nach den goldenen Regeln des Interface-Designs von Chandra & Guntupalli (2008)

Leitziel 1	Der Validierungsablauf erfolgt immer auf identische Weise.
Leitziel 2	Drag und Drop Funktionalität wird implementiert. Einstellungen werden visuell in das Interface verbaut.
Leitziel 3	Rückmeldungen werden über GitHub gesammelt.
Leitziel 4	Die Nutzer erhalten nur relevante Validierungsinformationen. Dialoge sind dynamisch ausklappbar zur Schlichthaltung der App.
Leitziel 5	Der Validierungsprozess kann nach Fehlerbehebungen einer Datei direkt weitergefahren werden.
Leitziel 6	Der Gesamtablauf kann zurückgesetzt werden.
Leitziel 7	Validierungs- und Darstellungsoptionen werden den Nutzern zur Verfügung gestellt.
Leitziel 8	Hilfehinweise erscheinen mit Hover- bzw. Schwebeeffekte.

Iterationsprozess

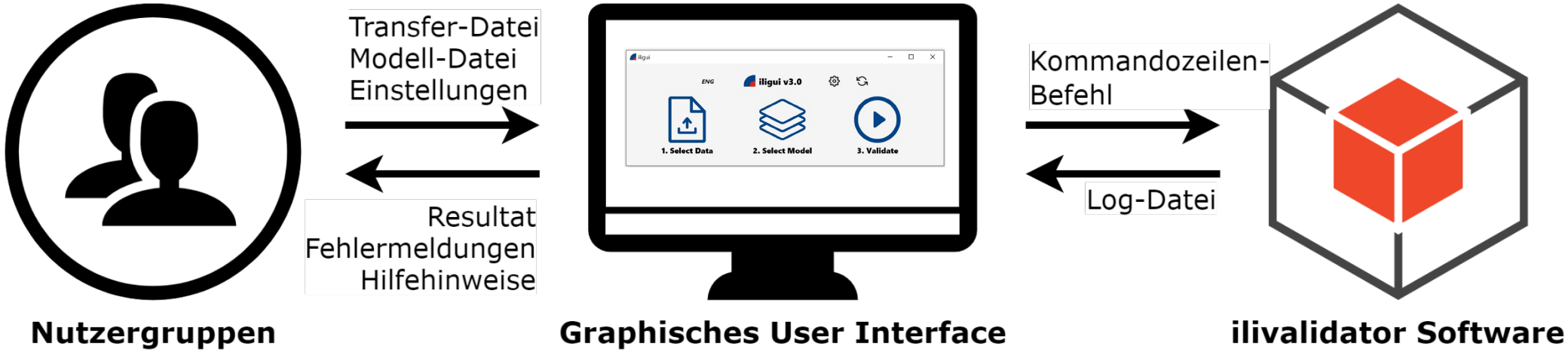
Fest definierte Iterationsschritte für die Entwicklung



Adaptiert von Blair-Early und Zender (2008)

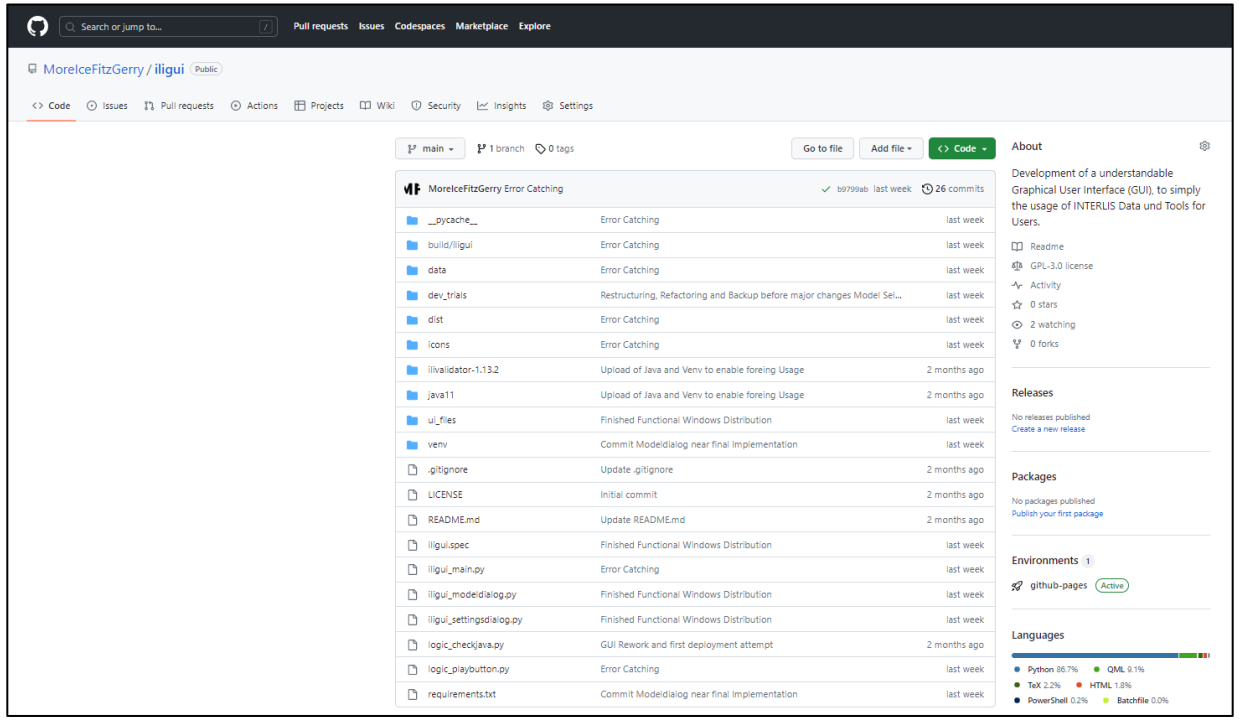
Software-Architektur

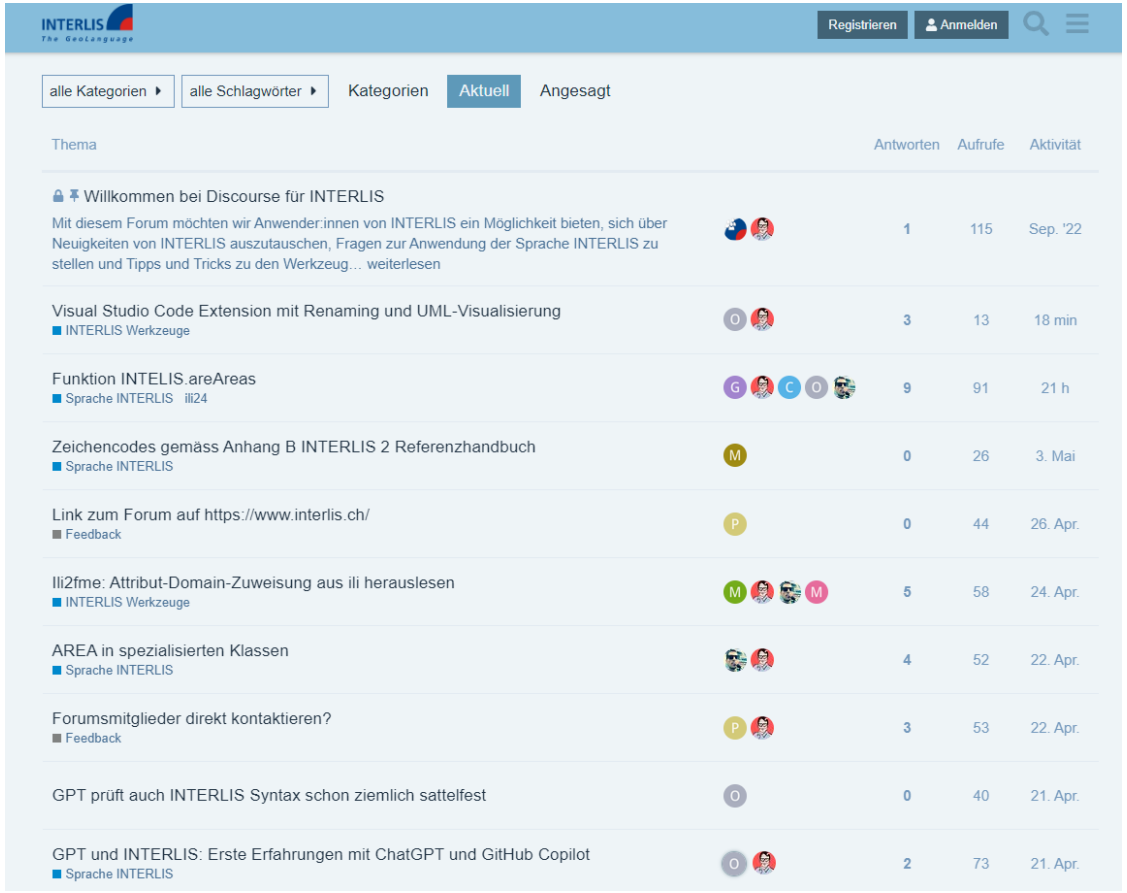
- Erstellen des GUI über den bestehenden ilivalidator. Die interne Software wird nicht verändert.
- Arbeit mit Python und PyQt6. Mit der Qt Designer Software bleibt das GUI auch zukünftig dynamisch anpassbar.
- Einbau der jeweils neusten ilivalidator Version und Java Runtime Environment.
- Distribution über PyInstaller Bibliothek auf dem jeweils erwünschten Betriebssystem.



Distribution und Aktualisierung

- Projekt Homepage über GitHub Pages veröffentlicht.
- Portables Executable für Windows und Unix zum Download verfügbar. MacOS folgt bei Nutzerbedarf.
- Gesamter Quellcode dokumentiert und offen auf GitHub unter ‘iligui’ erhältlich.
- Aktualisierungen werden auf der Homepage mitgeteilt und neu distribuiert.





Zukunft des iligui

Hilfehinweise für Fehler werden laufend ergänzt.
Das Projekt wird auf dem INTERLIS Forum publiziert.

Mögliche Weiterentwicklung:

- Übersetzungen auf Deutsch, Französisch, Spanisch
- Validierungen von mehreren Transferfiles im GUI
- Einstellungen via CONFIG.ini Files
- Einbau des INTERLIS Model Browser
- Zusätzliche Lösungsbeispiele zu den Fehlerhinweisen im GUI
- Automatisiertes Deployment über GitHub

Literatur

Chandra, Ravi und Guntupalli, Chaitanya (2008): *User Interface Design - Methods and Qualities of a Good User Interface Design*. [https://www.semanticscholar.org/paper/User-Interface-Design-Methods-and-Qualities-of-a-Chandra-Guntupalli/102c4ce13ba64c23b0c28018ac6d570661e91215; 3.6.2023].

Blair-Early, Adream und Zender, Mike (2008): *User Interface Design Principles for Interaction Design*. In: Design Issues 24/3. S. 85–107.

Dey, Pradip; Sinha, Bhaskar Raj; Amin, Mohammad und Badkoobehi, Hassan (2019): *Best Practices for Improving User Interface Design*. In: International Journal of Software Engineering & Applications 10 (September). S. 71–83. doi:10.5121/ijsea.2019.10505.

