kanton glarus

Bau und Umwelt
Raumentwicklung und Geoinformation
Kirchstrasse 2
8750 Glarus

WHITEPAPER

Thema **Generische Umsetzung der minimalen Geodatenmodelle in der kantonalen Geodaten-Infrastruktur**

Autor Fachstelle Geoinformation, Dr. Peter Staub

Publikation 2014-04-23

Der kantonale Raumdatenpool 1
Herausforderungen 1
- Methodisches 1
- Umsetzung objektorientierter Datenmodelle in relationalen Datenbanken 2
- Überführung der Produktivdaten in die Modellstruktur 3
- Assoziationen 3

Das Schnittstellenwerkzeug ili2pg 4
- Technologie 4
- Funktionsweise 4
- Anforderungen 4

Umsetzungsprozess 4
- Schritt 0 – Modelldefinition 5
- Schritt 1 – Datenmodell in der PostGIS-Datenbank anlegen 5
- Schritt 2 – Produktivdaten in die Modellstruktur umbauen 6
- Schritt 3 – Transferdaten exportieren 7
- Schritt 4 – Transferdaten prüfen und nutzen 7

Anwendungsbeispiele 7
- Kantonales Modell «Energieförderung» 7
- Bundesmodell «Stromversorgungssicherheit: Netzgebiete» (ID GeoIV 183.1) 10

Weiterführende Aspekte 12
- Andere Transferdatenformate 12
- Modellkonformer Austausch von Geodaten 12

Der kantonale Raumdatenpool

Die zentrale Datenhaltung in der kantonalen Geodaten-Infrastruktur wird als «Raumdatenpool» bezeichnet. Dabei kommen PostGIS-Datenbanken zum Einsatz, wobei wie üblich zwischen Test/Entwicklung/Integration und Produktion/Publikation unterschieden wird. Die Datennutzung erfolgt neben dem direkten Datenbankzugriff über Kartendienste (Datenviewer oder OGC WMS), Datendienste (OGC WFS) sowie als Datenbezug (Datei-Download).

Gängige Praxis bei der Datenintegration ist häufig immer noch die pragmatische 1:1-Überführung vorhandener, dateibasierter Datenbestände in die Datenbank, um die weitere Datenbearbeitung zu etablieren. Regelmässige Datenlieferungen aus externen Produktionssystemen erfolgen dateibasiert, aber häufig nicht modellbasiert. Im Zuge der Umsetzung des Einführungsgesetzes zum Geoinformationsgesetz (EG GeoIG) im Kanton Glarus müssen Geodatenmodelle entwickelt, adaptiert und nach den Regeln der Kunst umgesetzt werden. Dazu gehören die Datenbankkonfiguration aus konzeptionellen Datenmodellen, der Datenumbau aus den Produktivdaten in die Modellstruktur sowie der Datenexport und die Prüfung gegenüber dem Datenmodell.

Herausforderungen

Methodisches

Grundsätzlich ist es aus methodischer Sicht unerheblich, ob minimale Geodatenmodelle des Bundesrechts (MGDM) oder solche des kantonalen oder kommunalen Rechts (kMGDM) umzusetzen sind. Es wurde an anderer Stelle festgehalten, dass bei Geobasisdaten des Bundesrechts immer ausgehend vom MGDM als Basis etwaige kantonale oder kommunale Mehranforderungen in Form von Modellerweiterungen zu modellieren sind (Abb. 1).

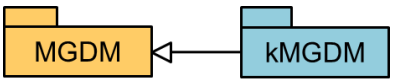


Abb. 1: Kantonale/kommunale Modelle als Erweiterung von Bundesmodellen

Bei der Realisierung wird schnell klar, dass die manuelle Umsetzung des umfangreichen Katalogs minimaler Geodatenmodelle nicht zielführend sein kann. Die Definition der nötigen Datenbankschemata ist aufwändig und kann trotzdem nicht überall eingesetzt werden, weil jede Geodaten-Infrastruktur ihre spezifischen Eigenheiten besitzt oder Konventionen unterworfen ist. Vielmehr muss es das Ziel sein, möglichst generische Schnittstellenwerkzeuge zu erhalten, welche die Modellumsetzung in der Datenbank erledigen (Abb. 2 und unten).

Abb. 2: Modellumsetzung mittels Schnittstellenwerkzeug «W»

Bei jedem Datenthema ist grundsätzlich zu entscheiden, ob die Datenproduktion auf die Modellstruktur umgestellt werden soll (Abb. 3, «A») oder ob weiterhin in der bestehenden Datenbankstruktur gearbeitet wird und die modellkonformen Daten bei Bedarf hergeleitet werden (Abb. 3, «B»).

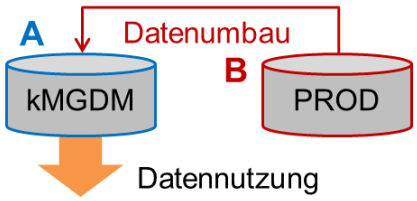


Abb. 3: Produktivdaten im Verhältnis zur Modellstruktur

Die Umstellung ist dann sinnvoll, wenn folgende Punkte mehrheitlich zutreffen:

- das bestehende Datenschema ist wenig umfangreich
- die bestehende Datenstruktur entspricht nicht mehr den Anforderungen
- es handelt sich um ein neues Thema, zu dem bislang noch keine Daten in der Datenbank bearbeitet und gepflegt wurden
- es hängen keine (komplexen) Anwendungen von diesen Daten ab
- Daten werden aus Drittsystemen mit einem Datenmodell geliefert.

Umsetzung objektorientierter Datenmodelle in relationalen Datenbanken

Es ist durchaus nachvollziehbar, dass die Umsetzung eher komplexerer Modellkonstrukte wie beispielsweise Vererbung, Strukturattribute oder etwa die Verwendung von CHBase-Katalogen in relationalen Datenbanken nicht offensichtlich ist. Aus technischer Sicht eröffnen sich zwei Möglichkeiten: entweder, man versucht mit umfangreichen SQL-Skripten die komplette Modellstruktur in der Datenbank abzubilden, oder man vertraut auf die Unterstützung geeigneter, generischer Schnittstellenwerkzeuge, um Datenmodelle umzusetzen.

Die erste Variante scheint zunächst bestechend, weil der Prozess der Abbildung objektorientierter Modellmerkmale in die relationale Datenbank transparent ist und jeder Schritt bewusst kontrolliert werden kann. Andererseits bedeutet dieses Vorgehen aber viel manuellen Konfigurationsaufwand bei gleichzeitiger hoher Fehleranfälligkeit was zu Inkonsistenzen über mehrere Datenmodelle hinweg führen kann. Die zweite Variante kompensiert diese Nachteile, weil die implementierten Abbildungsregeln bei jeder Anwendung exakt gleich ablaufen. Ein Datenmodell wird in Sekundenschnelle in die Datenbank überführt. Selbstverständlich muss dann vorausgesetzt werden können, dass das verwendete Schnittstellenwerkzeug alle Modellelemente gemäss Spezifikation umsetzen kann. Damit keine intransparenten Blackbox-Lösungen (s. Abb. 2) entstehen, sind OpenSource-Implementierungen mit Offenlegung des Quellcodes zu bevorzugen.

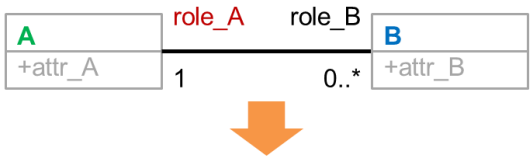
Überführung der Produktivdaten in die Modellstruktur

Eine weitere Hürde ist die Überführung der Produktivdaten in die Modellstruktur. Die historisch gewachsenen Datenschemata der Produktivdaten genügen den konzeptionellen Vorgaben der Datenmodelle nicht immer. Es ist also fallweise zu bestimmen, wie fehlende Modellteile aus dem Datenbestand herzuleiten oder eventuell neu zu erheben sind. Für den Datenumbau kommt das auf geografische Daten erweiterte Konzept *Extract–Transform–Load* (ETL) aus der Businesslogik zur Anwendung. Die verbreitetste und mächtigste ETL-Software für Geodaten ist die *Feature Manipulation Engine* (FME) der kanadischen Firma Safe Software Inc. Grundsätzlich könnte der Datenumbau auch direkt mittels SQL-Skripten realisiert werden, was sich allerdings Anbetracht der funktionalen Mächtigkeit und der nutzerfreundlichen Handhabung von FME in den meisten Fällen als nicht zweckmässig erweisen dürfte.

Assoziationen

Schliesslich bedarf die Umsetzung von Assoziationen der vertieften Betrachtung. Grundsätzlich können hierarchische Beziehungen mit der Einführung von Fremdschlüsselattributen verhältnismässig einfach realisiert werden (Abb. 4, «role_A»). Bei multiplen Beziehungen müssen Hilfstabellen zwischengeschaltet werden, die entsprechend zwei Fremdschlüssel enthalten.

Falls man in der Datenbank den vollständigen Gehalt von Assoziationen nutzen will – insbesondere dann, wenn die Datenproduktion auf die Modellstruktur umgestellt wird – muss das Fremdschlüsselattribut mittels Konsistenzbedingung als «Fremdschlüssel» deklariert werden. Die Beziehungsstärken «Assoziation», «Aggregation» und «Komposition» haben Einfluss auf das Objektverhalten bei Erzeugung, Änderung oder Löschung assoziierter Objekte.




```
CREATE TABLE A ( t_id …, attr_A … );

CREATE TABLE B ( t_id …, attr_B …, role_A …,
   CONSTRAINT role_A_fkey FOREIGN KEY (role_A)
   REFERENCES A (t_id) MATCH SIMPLE
   ON UPDATE CASCADE ON DELETE RESTRICT );
```

Abb. 4: Beziehung im Modell und auf der Datenbank

Das Schnittstellenwerkzeug ili2pg

Technologie

Das Schnittstellenwerkzeug *ili2pg* ist ein offenes Java-Programm, das von der Eisenhut Informatik AG entwickelt wird. Das Werkzeug verfügt über eine einfache grafische Benutzerschnittstelle, die aber nicht den vollständigen Funktionsumfang anbietet. Für die vollumfängliche Nutzung steht ein Kommandozeilenprogramm zur Verfügung. ili2pg ist systemneutral einsetzbar und im Stapelverarbeitungsmodus automatisierbar.

Funktionsweise

ili2pg besitzt drei Hauptfunktionalitäten:

- ein konzeptionelles INTERLIS-Datenmodell in der PostGIS-Datenbank anlegen
- INTERLIS-Transferdaten in die PostGIS-Datenbank laden
- INTERLIS-Transferdaten aus der PostGIS-Datenbank exportieren.

Für alle Funktionalitäten steht eine Reihe von Kommandozeilenoptionen zur Verfügung. So kann beispielsweise ein SQL-Skript erzeugt werden, das die Schema- und Tabellendefinition gemäss Modell enthält. Alle Funktionalitäten sind sowohl für INTERLIS 1 als auch für INTERLIS 2.3 nutzbar.

Anforderungen

Im Kontext der Umsetzung von minimalen Geodatenmodellen muss das Schnittstellenwerkzeug ili2pg insbesondere folgende Modellkonstrukte unterstützen: Wertebereichsdefinitionen, Strukturen und Strukturattribute, Assoziationen, Referenzattribute, Vererbung, den Umgang mit Datenmodellablagen und mit den CHBase-Modulen.

Umsetzungsprozess

Die Umsetzung der minimalen Geodatenmodelle im kantonalen Raumdatenpool ist als Prozess in fünf Schritten definiert (Abb. 5):

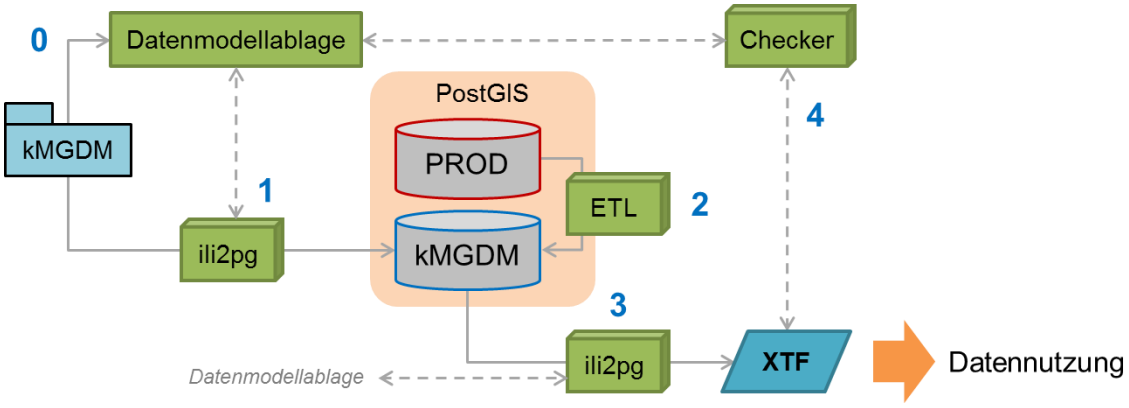


Abb. 5: Umsetzungsprozess

Schritt 0 – Modelldefinition

Die Definition der minimalen Geodatenmodelle ist an zahlreichen anderen Stellen ausführlich beschrieben und wird an dieser Stelle lediglich der konzeptionellen Vollständigkeit halber aufgeführt. Wesentlich ist die Publikation der konzeptionellen Datenmodelle in der Datenmodellablage des Bundes beziehungsweise des Kantons (kMGDM). Dadurch können die Datenmodelle von den eingesetzten Schnittstellen-, Datenumbau- und Prüfwerkzeugen genutzt werden.

In Abb. 6 ist ein abstraktes Datenmodellbeispiel dargestellt, das in den nachfolgenden Erläuterungen zur Illustration dient.

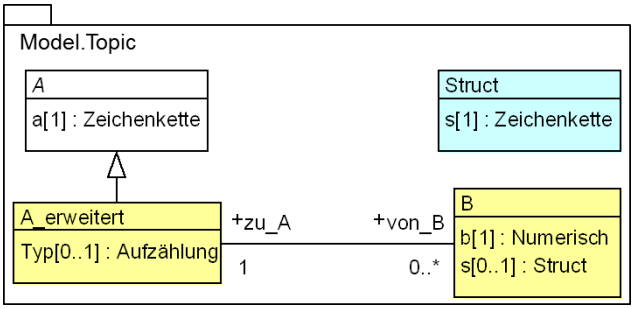


Abb. 6: Fiktives Datenmodellbeispiel

Schritt 1 – Datenmodell in der PostGIS-Datenbank anlegen

Mit dem Schnittstellenwerkzeug ili2pg wird die INTERLIS 2.3-Modelldefinition aus der Datenmodellablage in die PostGIS-Datenbank geladen und als neues Datenschema konfiguriert. Das Programm setzt die Modellelemente wie folgt um:

- Klassen und Strukturen: Pro Objekt eine Tabelle, Einführung Tabellenidentifikator `T_Id` als Primärschlüssel plus Sequenzattribut `T_Seq` für die Reihenfolge der Strukturelemente.

```
CREATE TABLE ….B (
  T_Id integer PRIMARY KEY,
  b decimal(10,3) NOT NULL,
  zu_A integer NULL
);
```

```
CREATE TABLE ….Struct (
  T_Id integer PRIMARY KEY,
  T_Seq integer NOT NULL,
  s varchar(100) NOT NULL,
  modeltopic_B_s integer NULL
);
```

- Vererbung: Jede Klasse wird in eine Tabelle abgebildet, was die Aufteilung der Objektinstanzen auf mehrere Tabellen zur Folge hat (sog. *NewClass*-Strategie). Die Oberklasse erhält jeweils ein Attribut `T_Type`, das den Klassennamen der Unterklasse enthält.

```
CREATE TABLE ….A (
  T_Id integer PRIMARY KEY,
  T_Type varchar(60) NOT NULL,
  a varchar(24) NOT NULL
);
```

```
CREATE TABLE ….A_erweitert (
  T_Id integer PRIMARY KEY,
  Typ integer NULL
);
```

- Assoziationen und Referenzattribute: Einführung von Fremdschlüsselattributen, jedoch ohne automatische Erzeugung der oben beschriebenen Konsistenzbedingungen für Fremdschlüssel.

```
CREATE TABLE ….A_erweitert (
  T_Id integer PRIMARY KEY,
  Typ integer NULL
);
```

ggf. manuell nachkonfigurieren!

```
CREATE TABLE ….B (
  T_Id integer PRIMARY KEY,
  b decimal(10,3) NOT NULL,
  zu_A integer NULL,
  CONSTRAINT zu_A_fkey FOREIGN KEY (zu_A)
  REFERENCES A (T_Id) MATCH SIMPLE
  ON UPDATE CASCADE ON DELETE RESTRICT
);
```

- Strukturattribute: Erzeugung eines Fremdschlüsselattributs in der Struktur-Tabelle. Der Name des Attributs wird aus dem Klassen- und dem Strukturattribut-Namen gebildet.

```
CREATE TABLE ….B (
  T_Id integer PRIMARY KEY,
  b decimal(10,3) NOT NULL,
  zu_A integer NULL
);
```

```
CREATE TABLE ….Struct (
  T_Id integer PRIMARY KEY,
  T_Seq integer NOT NULL,
  s varchar(100) NOT NULL,
  modeltopic_B_s integer NULL
);
```

- Wertebereichsdefinitionen: Optionale Erzeugung von Lookup-Tabellen (folgende Abb.) beziehungsweise Import der Aufzählungselemente als zusätzliche Textspalte.

```
CREATE TABLE ….A_erweitert (
  T_Id integer PRIMARY KEY,
  Typ integer NULL
);
```

```
CREATE TABLE ….A_erweitert_Typ (
  itfCode integer PRIMARY KEY,
  iliCode varchar(1024) NOT NULL,
  seq integer NULL,
  dispName varchar(250) NOT NULL
);
```

Aufzählung gem. Modell
Reihenfolge (optional)

- Zusätzlich werden einige Systemtabellen angelegt, die ili2pg vor allem für den Datenexport benötigt. Diese Tabellen müssen nicht aktiv bewirtschaftet werden.

Schritt 2 – Produktivdaten in die Modellstruktur umbauen

Beim Datenumbau werden die Objektinstanzen aus der Produktivdatenbank in die Modellstruktur überführt. Wenn die Produktion auf die Modellstruktur umgestellt wird, muss dieser Schritt nur einmal ausgeführt werden. Falls weiterhin in der gegebenen Produktivdatenbank gearbeitet wird, so ist der Datenumbau bei Bedarf, regelmässig oder sogar automatisiert durchzuführen. Der Datenumbau findet innerhalb der PostGIS-Datenbank statt.

Im kantonalen Raumdatenpool wird die ETL-Software FME für den Datenumbau verwendet. Pro Datenmodell werden die Tabellen aus dem Produktivschema mit dem PostGIS-Reader geladen und via PostGIS-Writer werden die Tabellen gemäss Modellschema bereitgestellt. Die nötigen Datenumbauschritte werden mit Hilfe von FME-Transformatoren konfiguriert.

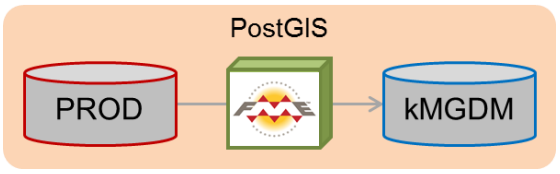


Abb. 7: Datenumbau mit FME

Der fertig konfigurierte Datenumbau mittels FME kann im Stapelverarbeitungsmodus automatisiert werden (vgl. auch FME Server).

Schritt 3 – Transferdaten exportieren

Die modellkonformen PostGIS-Daten werden mit ili2pg als INTERLIS 2-XML-Transferformat exportiert. Die Modellelemente werden gemäss INTERLIS 2.3 Referenzhandbuch, Kapitel 3, kodiert.

Schritt 4 – Transferdaten prüfen und nutzen

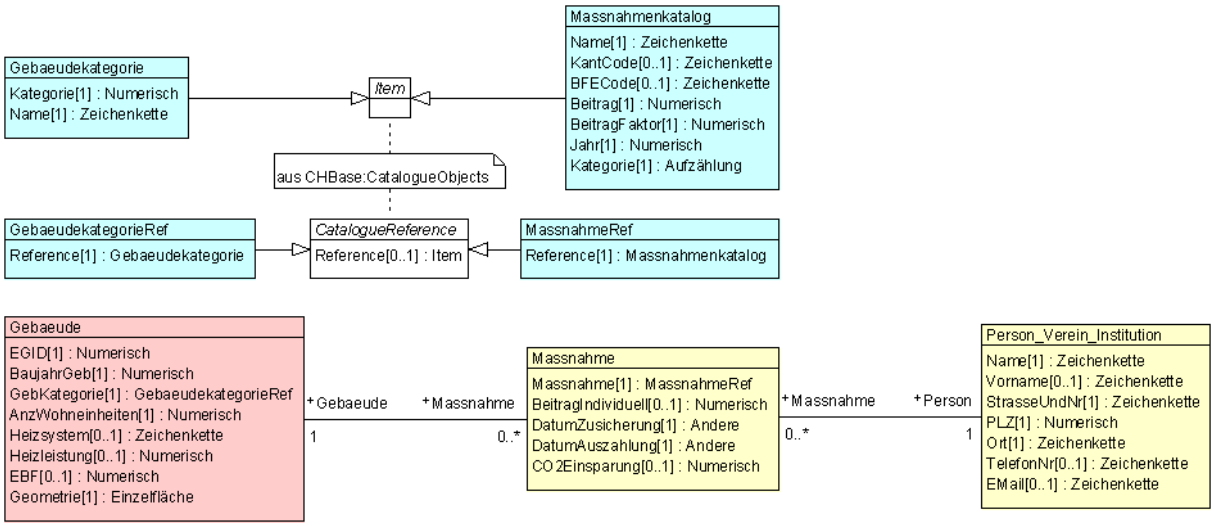
Um die Qualität der Exportdaten sicherzustellen, wird der exportierte Datensatz im *Checker für INTERLIS 2* gegenüber dem Datenmodell geprüft. Bei dieser Prüfung wird nicht nur die Syntax überprüft, sondern auch der Inhalt: kommen ungültige Aufzählungswerte vor? Stimmt die Geometrie? Sind alle Beziehungsreferenzen korrekt? Nach erfolgreicher Prüfung können die Daten weitergegeben und genutzt werden.

Anwendungsbeispiele

Im Verlauf der Weiterentwicklung von ili2pg 2014/15 und im Rahmen eines kantonsinternen Projekts sowie in Zusammenarbeit mit dem Bundesamt für Energie BFE und der Eidgenössischen Elektrizitätskommission ElCom sind im Frühjahr 2015 zwei Datenmodelle vollständig umgesetzt worden.

Kantonales Modell «Energieförderung»

Schritt 0 – UML-Datenmodell:



Schritt 1 – ili2pg-Befehl zur PostGIS-Konfiguration mit dem Datenmodell:

```
java -jar ili2pg.jar --schemaimport --dbhost MY_HOST --dbport 5432
   --dbusr MY_USER --dbpwd MY_PASSWORD --dbdatabase MY_DATABASE
   --dbschema kmgdm_energiefoerderung --createscript kmgdm_efo.sql
   --createGeomIdx --strokeArcs Energiefoerderung_V1.ili
```

Schritt 2 – Datenumbau mit FME: nicht anwendbar, da neue Daten originär in der Modellstruktur erfasst werden. Dafür wird für dieses Beispiel die Attributbildung auf der PostGIS-Datenbank dargestellt:

Item	
t_id	Laufender Tabellenidentifikator
t_type	SQL-Name der erweiterten Klasse/Struktur: «Gebaeudekategorie» bzw. «Massnahmenkatalog»

CatalogueReference	
t_id	Laufender Tabellenidentifikator
t_type	SQL-Name der erweiterten Klasse/Struktur: «GebaeudekategorieRef» bzw. «MassnahmeRef»
t_seq	Reihenfolge der Strukturelemente – hier ohne Belang, weil nur eines vorkommt; muss aber gesetzt werden
Reference	Referenz auf das verwendete Katalogelement: Fremdschlüssel → Gebaeudekategorie bzw. → Massnahmenkatalog
energrfng_v1energiefordrun_gebaeude_gebkategorie	Erzeugter Fremdschlüssel → Gebaeude (aus dem Strukturattribut GebKategorie in der Klasse Gebaeude)
energrfng_v1energiefordrun_massnahme_massnahme	Erzeugter Fremdschlüssel → Massnahme (aus dem Strukturattibut «Massnahme» in der Klasse Massnahme)

Gebaeudekategorie	**(Erweiterung von Item; analog für Massnahmenkatalog!)**
t_id	Erzeugter Fremdschlüssel → Item (aus der Vererbung)
(Klassenattribute)	–

GebaeudekategorieRef	**(Erweiterung von CatalogueReference; analog für MassnahmeRef!)**
t_id	Erzeugter Fremdschlüssel → CatalogueReference (aus der Vererbung)
(Klassenattribute)	–

Gebaeude	
t_id	Laufender Tabellenidentifikator
(Klassenattribute)	–

Person_Verein_Institution	
t_id	Laufender Tabellenidentifikator
(Klassenattribute)	–

Schritt 3 – ili2pg-Befehl für den Datenexport im INTERLIS 2-XML-Transferformat:

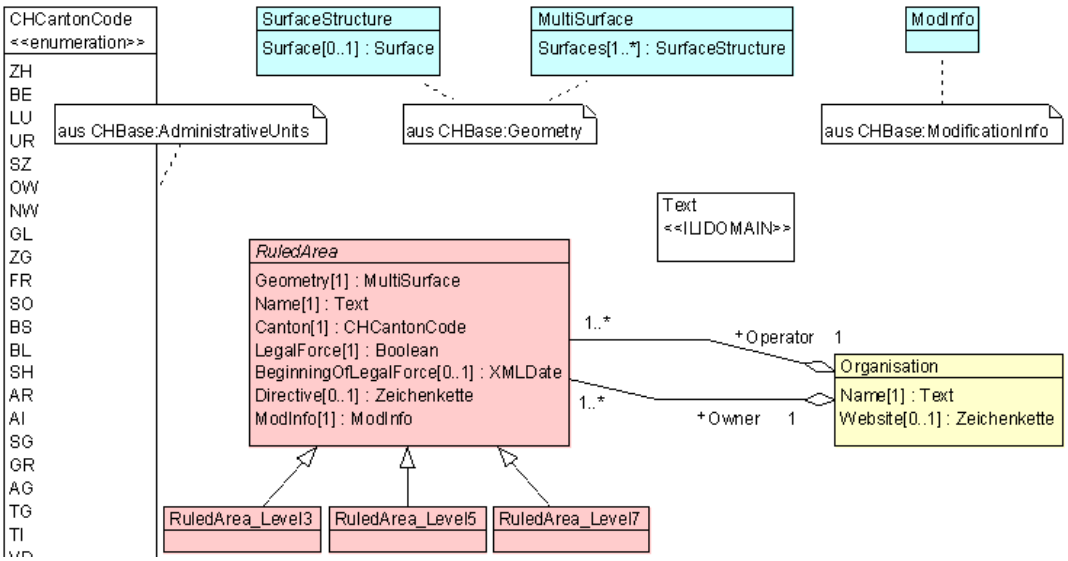
```
java -jar ili2pg.jar --export --dbhost MY_HOST --dbport 5432
   --dbusr MY_USER --dbpwd MY_PASSWORD --dbdatabase MY_DATABASE
   --dbschema kmgdm_energiefoerderung --models Energiefoerderung_V1
   energiefoerderung.xtf
```

Struktur der Exportdatei:


```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSFER xmlns="http://www.interlis.ch/INTERLIS2.3">
<HEADERSECTION SENDER="ili2pg-2.1.1-20150410" VERSION="2.3">

  <MODELS>
    <MODEL NAME="CatalogueObjects_V1" VERSION="2011-08-30" URI="http://www.geo.admin.ch"></MODEL>
    <MODEL NAME="CatalogueObjectTrees_V1" VERSION="2011-08-30" URI="http://www.geo.admin.ch"></MODEL>
    <MODEL NAME="CoordSys" VERSION="2005-06-16" URI="http://www.interlis.ch/models"></MODEL>
    <MODEL NAME="Units" VERSION="2012-02-20" URI="http://www.interlis.ch/models"></MODEL>
    <MODEL NAME="GeometryCHLV03_V1" VERSION="2015-02-20" URI="http://www.geo.admin.ch"></MODEL>
    <MODEL NAME="GeometryCHLV95_V1" VERSION="2015-02-20" URI="http://www.geo.admin.ch"></MODEL>
    <MODEL NAME="Energiefoerderung_V1" VERSION="2014-11-30" URI="http://www.geo.gl.ch"></MODEL>
  </MODELS>

</HEADERSECTION>
<DATASECTION>

  <Energiefoerderung_V1.Energiefoerderung_Kataloge BID="Energiefoerderung_V1.Energiefoerderung_Kataloge">
    <Energiefoerderung_V1.Energiefoerderung_Kataloge.Gebaeudekategorie TID="10001">
      <Kategorie>1</Kategorie>
      <Name>Wohnen Mehrfamilienhaus</Name>
    </Energiefoerderung_V1.Energiefoerderung_Kataloge.Gebaeudekategorie>
  </Energiefoerderung_V1.Energiefoerderung_Kataloge>

  <Energiefoerderung_V1.Energiefoerderung BID="Energiefoerderung_V1.Energiefoerderung">
    <Energiefoerderung_V1.Energiefoerderung.Gebaeude TID="9990001">
      <EGID>123456</EGID>
      <BaujahrGeb>2011</BaujahrGeb>
      <GebKategorie>
        <Energiefoerderung_V1.Energiefoerderung_Kataloge.GebaeudekategorieRef>
          <Reference REF="10001" BID="Energiefoerderung_V1.Energiefoerderung_Kataloge"></Reference>
        </Energiefoerderung_V1.Energiefoerderung_Kataloge.GebaeudekategorieRef>
      </GebKategorie>
      <AnzWohneinheiten>7</AnzWohneinheiten>
      <Geometrie>
        <SURFACE> ... </SURFACE>
      </Geometrie>
    </Energiefoerderung_V1.Energiefoerderung.Gebaeude>
  </Energiefoerderung_V1.Energiefoerderung>

</DATASECTION>
</TRANSFER>
```

Bundesmodell «Stromversorgungssicherheit: Netzgebiete» (ID GeoIV 183.1)

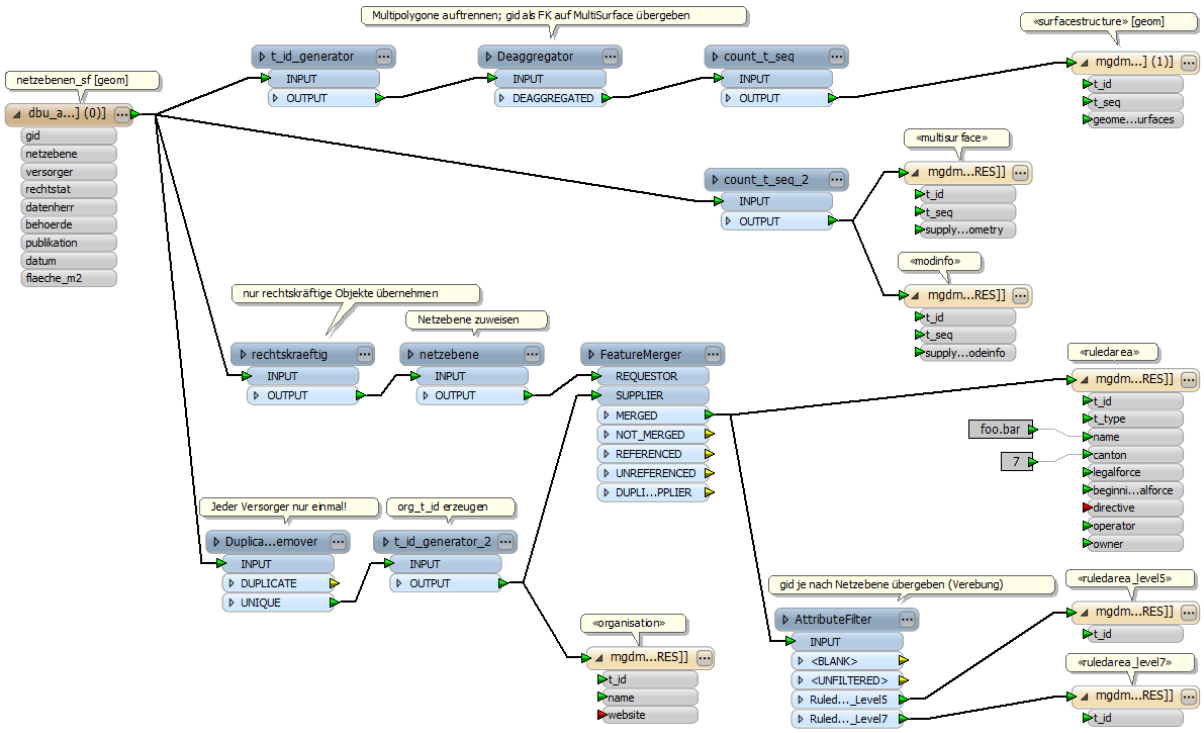
Schritt 0 – UML-Datenmodell:



Schritt 1 – ili2pg-Befehl zur PostGIS-Konfiguration mit dem Datenmodell:

```
java -jar ili2pg.jar --schemaimport --dbhost MY_HOST --dbport 5432
   --dbusr MY_USER --dbpwd MY_PASSWORD --dbdatabase MY_DATABASE
   --dbschema mgdm_supplysecurity --createscript mgdm_supplsec.sql
   --createGeomIdx --strokeArcs SupplySecurity_RuledAreas_V1.ili
```

Schritt 2 – Datenumbau mit FME:



Schritt 3 – ili2pg-Befehl für den Datenexport im INTERLIS 2-XML-Transferformat:

```
java -jar ili2pg.jar --export --dbhost MY_HOST --dbport 5432
   --dbusr MY_USER --dbpwd MY_PASSWORD --dbdatabase MY_DATABASE
   --dbschema mgdm_supplysecurity --models SupplySecurity_RuledAreas_V1
   supplysecurity.xtf
```

Struktur der Exportdatei:


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

<HEADERSECTION SENDER="ili2pg-2.1.1-20150410" VERSION="2.3">
  <MODELS>
    ... (alle importierten Modelle; v.a. aus Repositories)
    <MODEL NAME="SupplySecurity_RuledAreas_V1" VERSION="2015-04-02« URI="mailto:geoinformation@localhost"></MODEL>
  </MODELS>
</HEADERSECTION>

<DATASECTION>
  <SupplySecurity_RuledAreas_V1.SupplySecurity_RuledAreas_WithOneState BID="SupplySecurity_RuledAreas_V1.SupplySecu
    <SupplySecurity_RuledAreas_V1.SupplySecurity_RuledAreas_WithOneState.Organisation TID="0">
      <Name>Axpo / TBGS</Name>
    </SupplySecurity_RuledAreas_V1.SupplySecurity_RuledAreas_WithOneState.Organisation>
    ...
    <SupplySecurity_RuledAreas_V1.SupplySecurity_RuledAreas_WithOneState.RuledArea_Level5 TID="143">
      <Geometry>
        <GeometryCHLV03_V1.MultiSurface>
          <Surfaces>
            <GeometryCHLV03_V1.SurfaceStructure>
              <Surface>
                <SURFACE><BOUNDARY><POLYLINE><COORD><C1>729334.551</C1><C2>221299.017</C2></COORD>...
              </Surface>
            </GeometryCHLV03_V1.SurfaceStructure>
          </Surfaces>
        </GeometryCHLV03_V1.MultiSurface>
      </Geometry>
      <Name>foo.bar</Name>
      <Canton>GL</Canton>
      <LegalForce>true</LegalForce>
      <BeginningOfLegalForce>2014-01-07</BeginningOfLegalForce>
      <ModeInfo>
        <WithOneState_V1.ModInfo></WithOneState_V1.ModInfo>
      </ModeInfo>
      <Operator REF="0"></Operator>
      <Owner REF="0"></Owner>
    </SupplySecurity_RuledAreas_V1.SupplySecurity_RuledAreas_WithOneState.RuledArea_Level5>
    ... (inkl. RuledArea_Level7)
  </SupplySecurity_RuledAreas_V1.SupplySecurity_RuledAreas_WithOneState>
</DATASECTION>

</TRANSFER>
```

Weiterführende Aspekte

Andere Transferdatenformate

Für den in der Einleitung erwähnten Datenbezug dürften nur in zweiter Linie INTERLIS 2-XML-Daten nachgefragt werden. In Sinne einer nutzerorientierten Datenbereitstellung müssen weitere Abgabeformate aus den modellkonformen Daten abgeleitet werden. Dazu können Formate wie GeoPackage, GML oder auch CSV gehören.

FME oder offene Bibliotheken wie ogr2ogr bieten einfache 1:1-Prozessoren für die Umformatierung an, die für diese Aufgabe eingesetzt werden können. Alternativ können einfache Transferformate auch ohne «Umweg» über das INTERLIS 2-XML-Transferformat direkt aus PostGIS exportiert werden.

Modellkonformer Austausch von Geodaten

Das Projekt «Modellkonformer Austausch von Geodaten» (MDX) in Zusammenarbeit von IKGEO und GKG/KOGIS erörtert die Bedeutung des Begriffs «Download-Dienst» im Sinne der Geoinformationsverordnung (GeoIV) sowie die technischen Voraussetzungen und Möglichkeiten, solche Download-Dienste modellkonform zu realisieren.

In diesem Zusammenhang steht neben dem INTERLIS 2-XML-Transferformat auch GML gemäss *eCH-0118 GML-Kodierungsregeln für INTERLIS* im Vordergrund, insbesondere bei der Umsetzung der Download-Dienste als OGC WFS. Um diese Variante optimal zu unterstützen, soll das Schnittstellenwerkzeug ili2pg erweitert werden, damit eCH-0118-konforme GML-Transferdaten aus PostGIS exportiert werden können.